Robot Autonomy

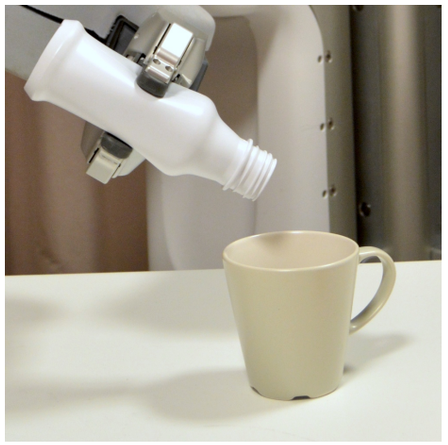
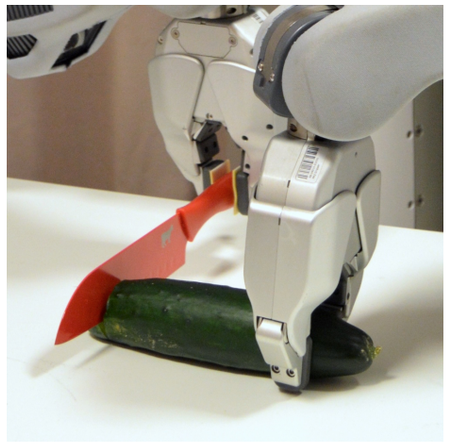
# Lecture 3:
# Self-supervised Learning

Oliver Kroemer

# Motivation

- Robots need to understand their environment

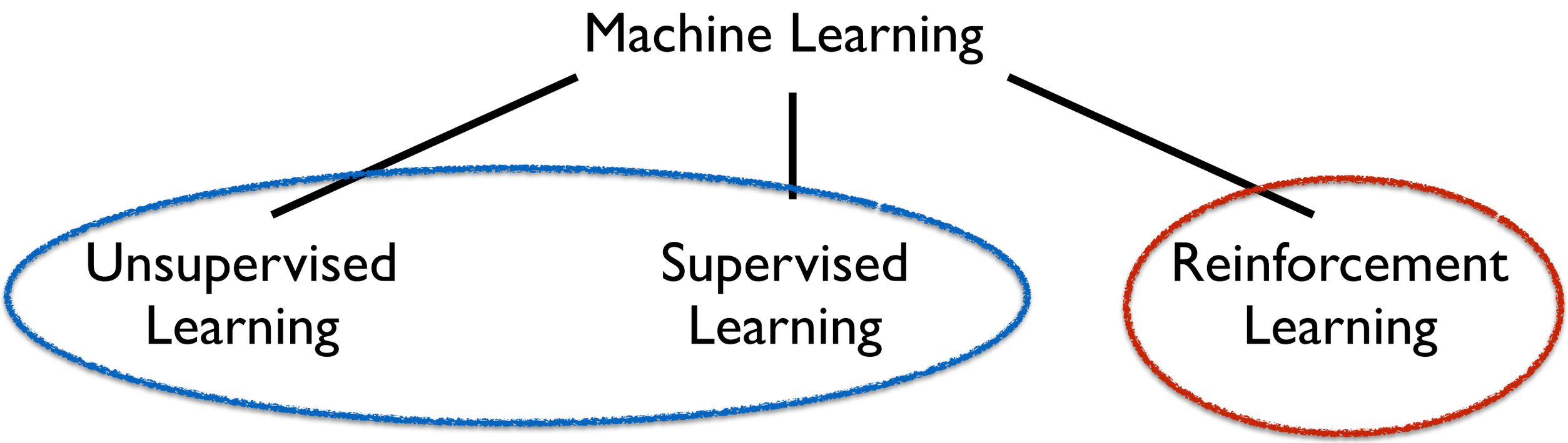


  - Identify object properties and action possibilities
  - Predict the effects of different interactions
  - Use this information to perform tasks

- Robot to increase autonomy by learning on its own

# Machine Learning



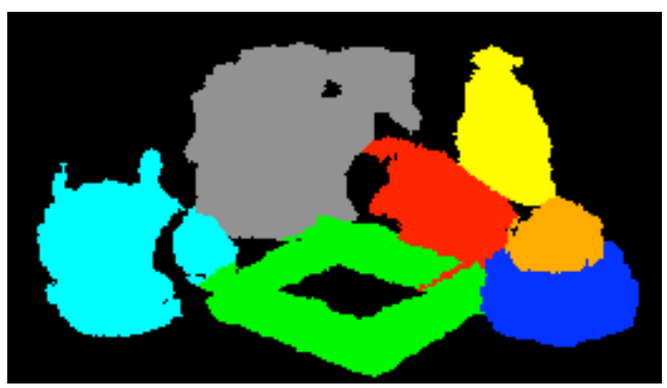

**Today**

- Robots can acquire information from many sources
- Will be focusing on learning from own experiences

# Interactive Perception

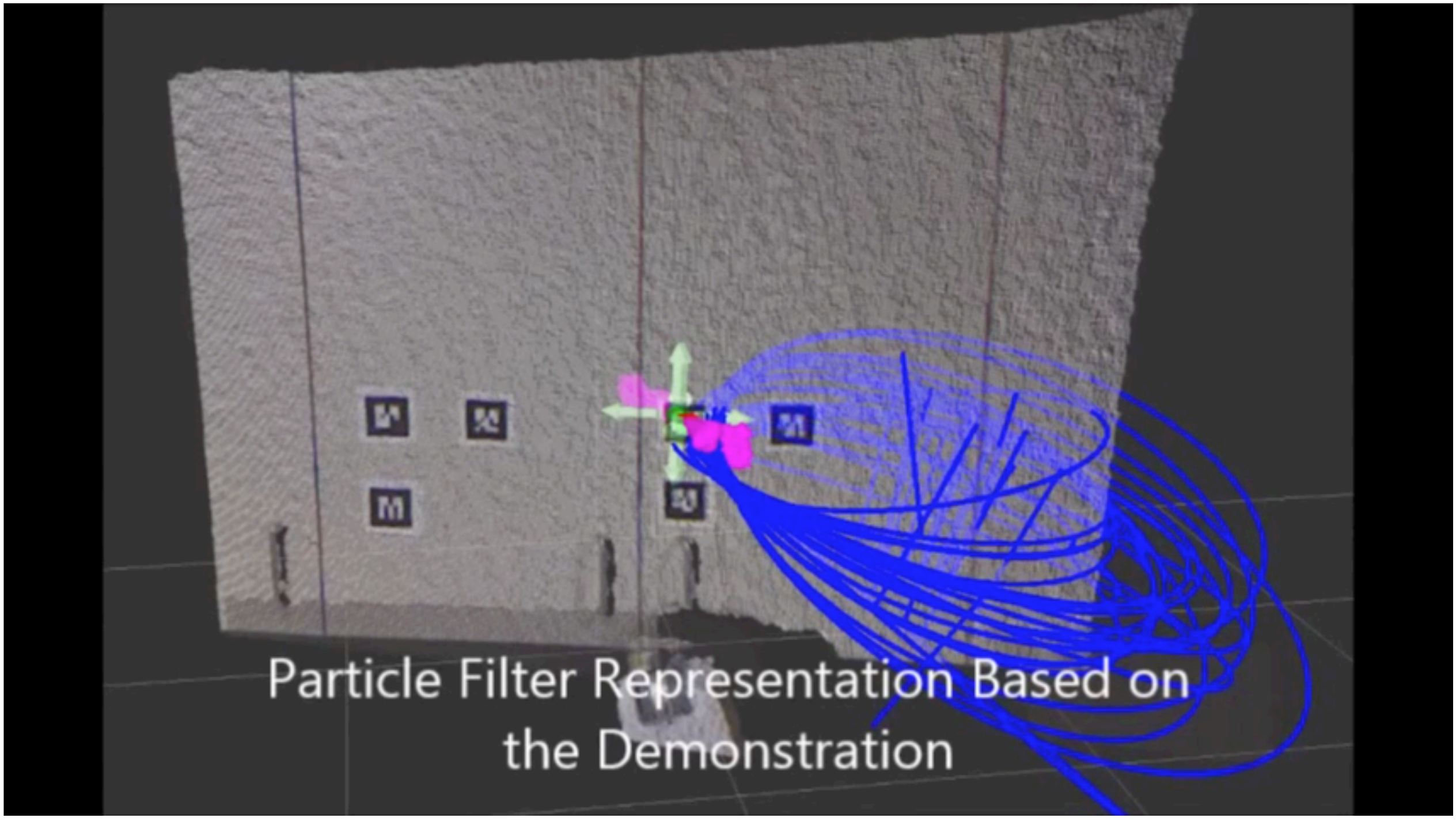
- Robot can perform actions to acquire more information
  - Change point of view to get a different perspective
  - Perform actions and observe effects - interactive perception
- Interactive perception more informative than passive
  - Additional sensor modalities, e.g., tactile
  - Infer latent properties of environment, e.g., mass
  - Overcome visual ambiguity
- Additional information often basis for self-supervised learning

# Segmentation Example

van Hoof et al. Probabilistic Segmentation and Targeted Exploration of Objects in Cluttered Environments TRo 2014

# Kinematics Example




Hausman et al. Active Articulation Model Estimation through Interactive Perception ICRA 2015

# Material Property Example

Sinapov et al. Grounding Semantic Categories in Behavioral Interactions: Experiments with 100 Objects RAS 2014

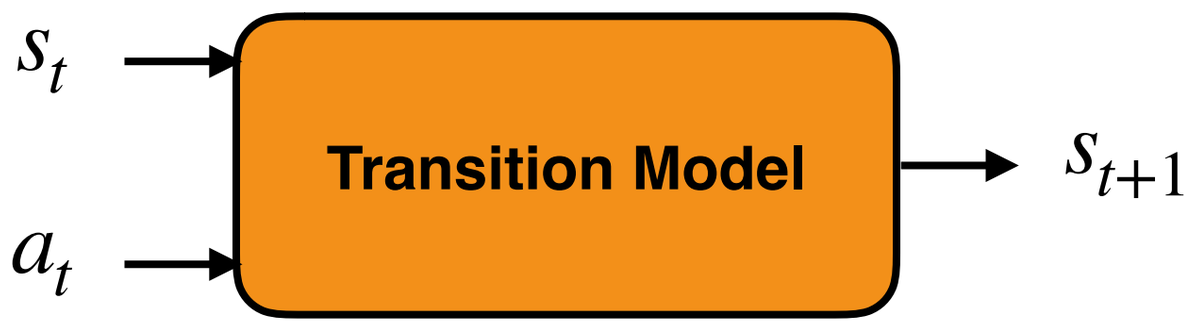
# State Transition Models

# Transition Models

- Transition probability

$$p(s_{t+1} \mid a_t, s_t)$$

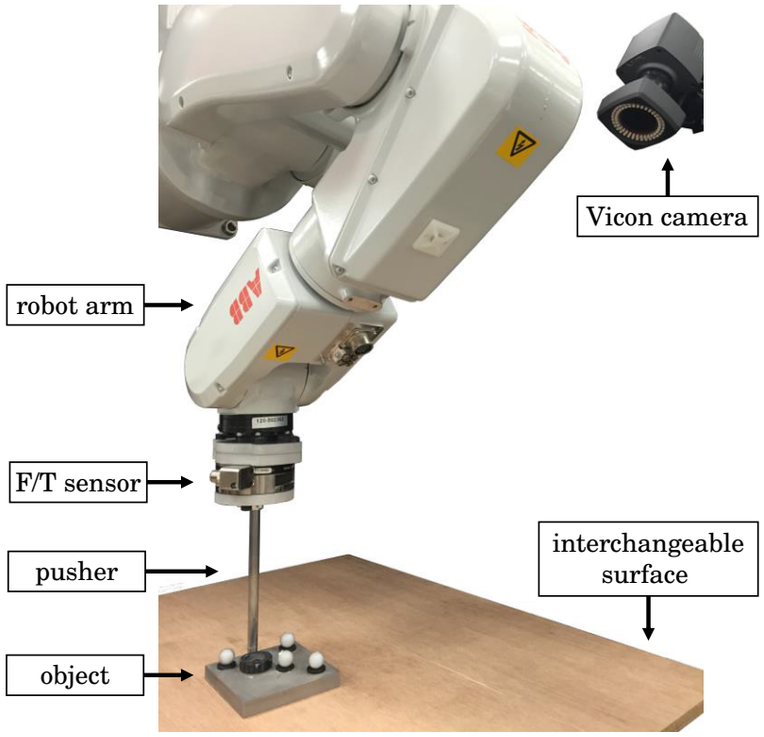
- Regression problem for continuous states

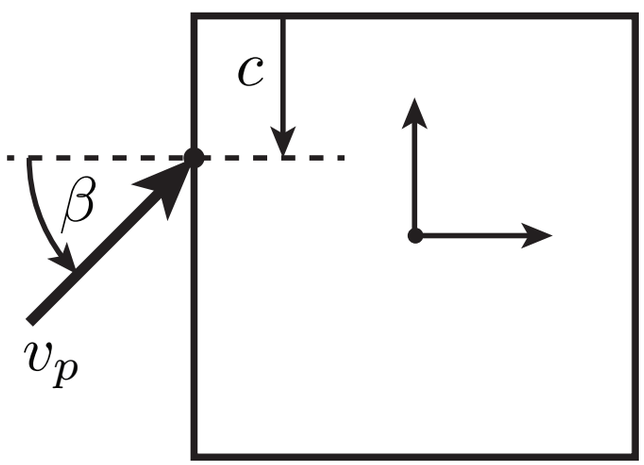


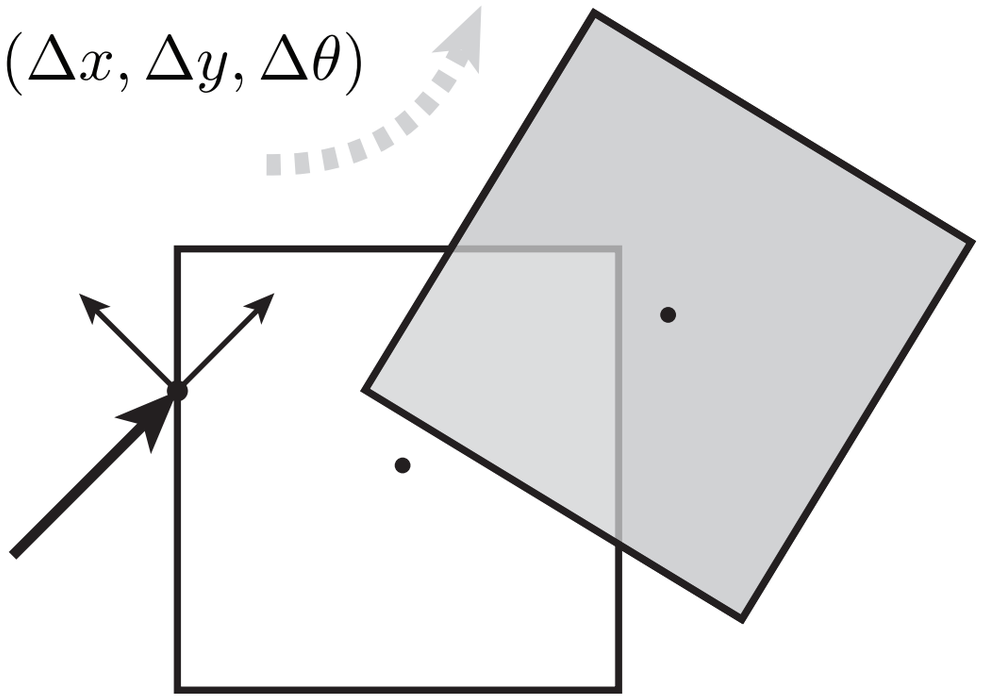

- Robot can acquire these training samples autonomously
  - Observe state, perform an action, and observe the next state
  - Samples can be acquired independent of a specific task
- Capture transitions using various different models
  - Linear regression, Gaussian process regression, neural networks,…

# Model for Pushing

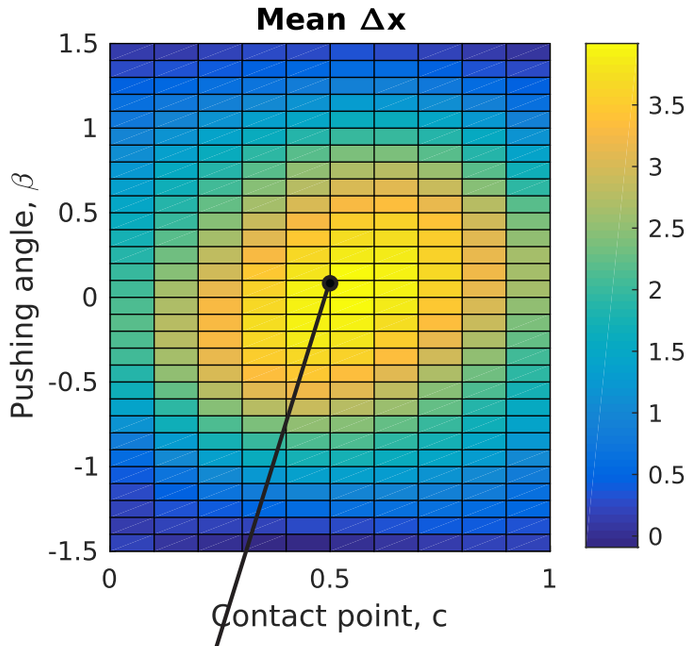
- Use a Gaussian Process to model planar pushing effects
  - Input: Contact point position, angle, and velocity
  - Output: contact-frame-relative shifts in position and orientation

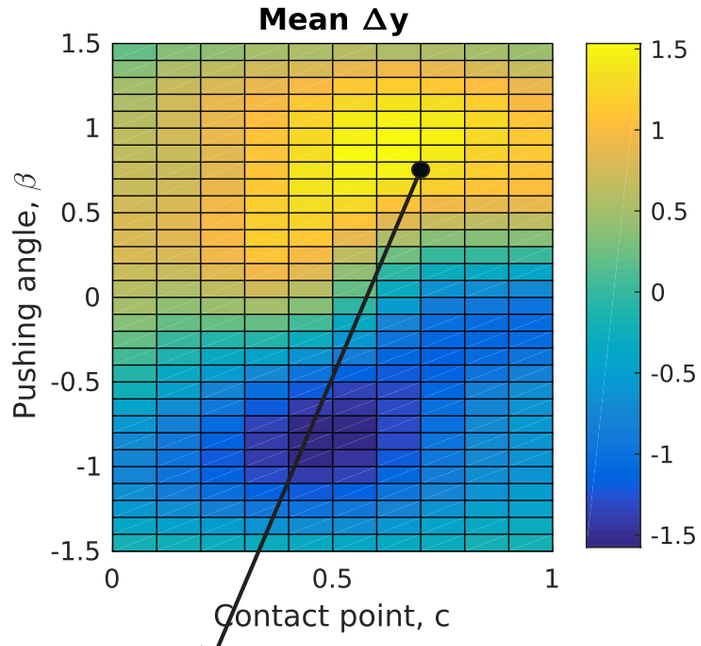


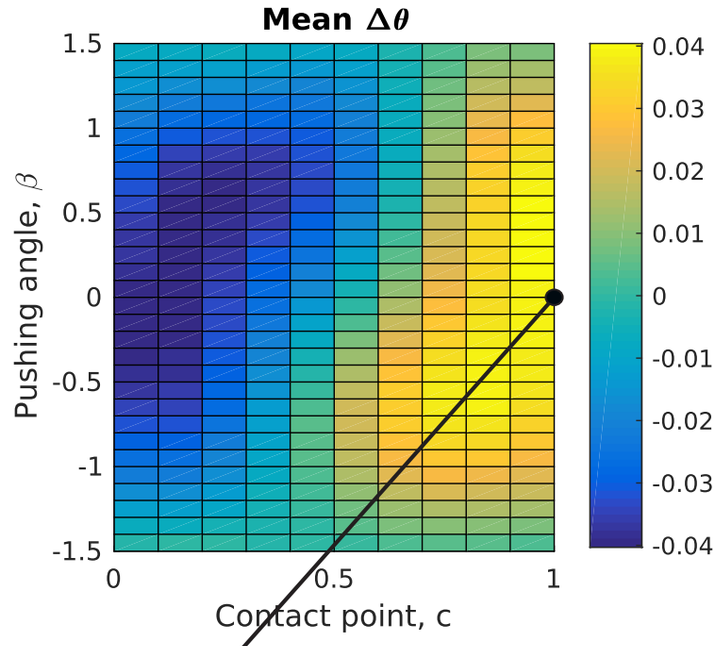



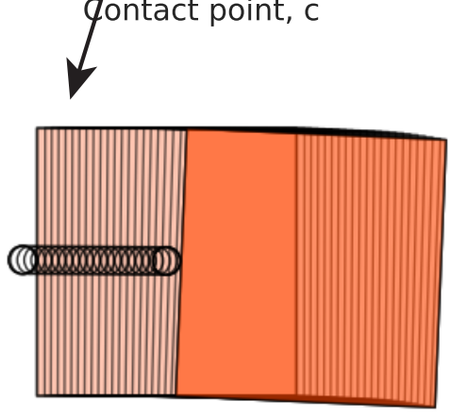



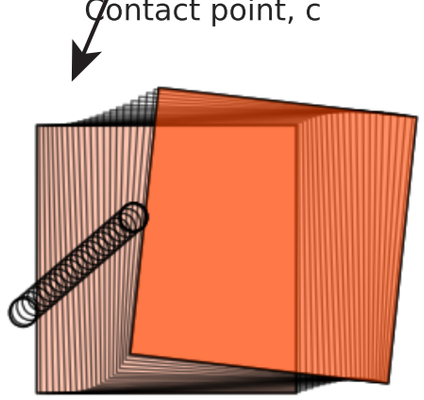
# Model for Pushing

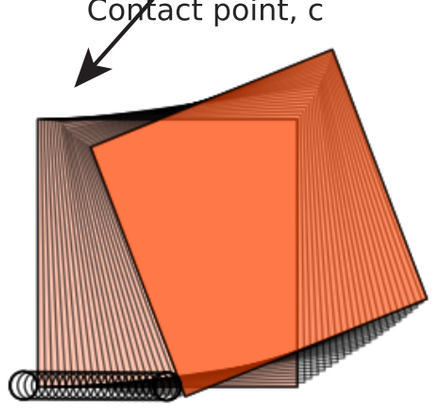


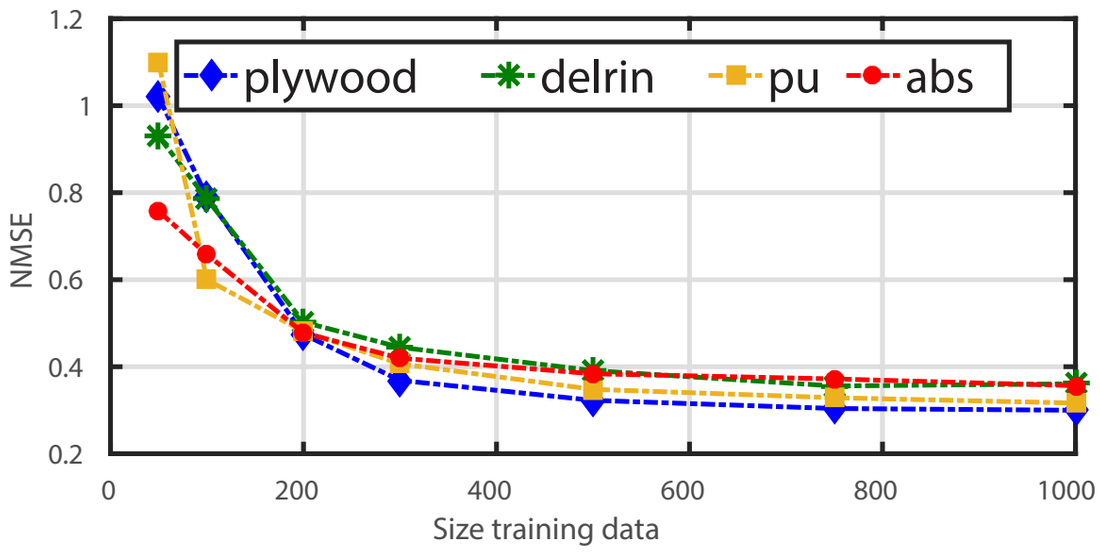



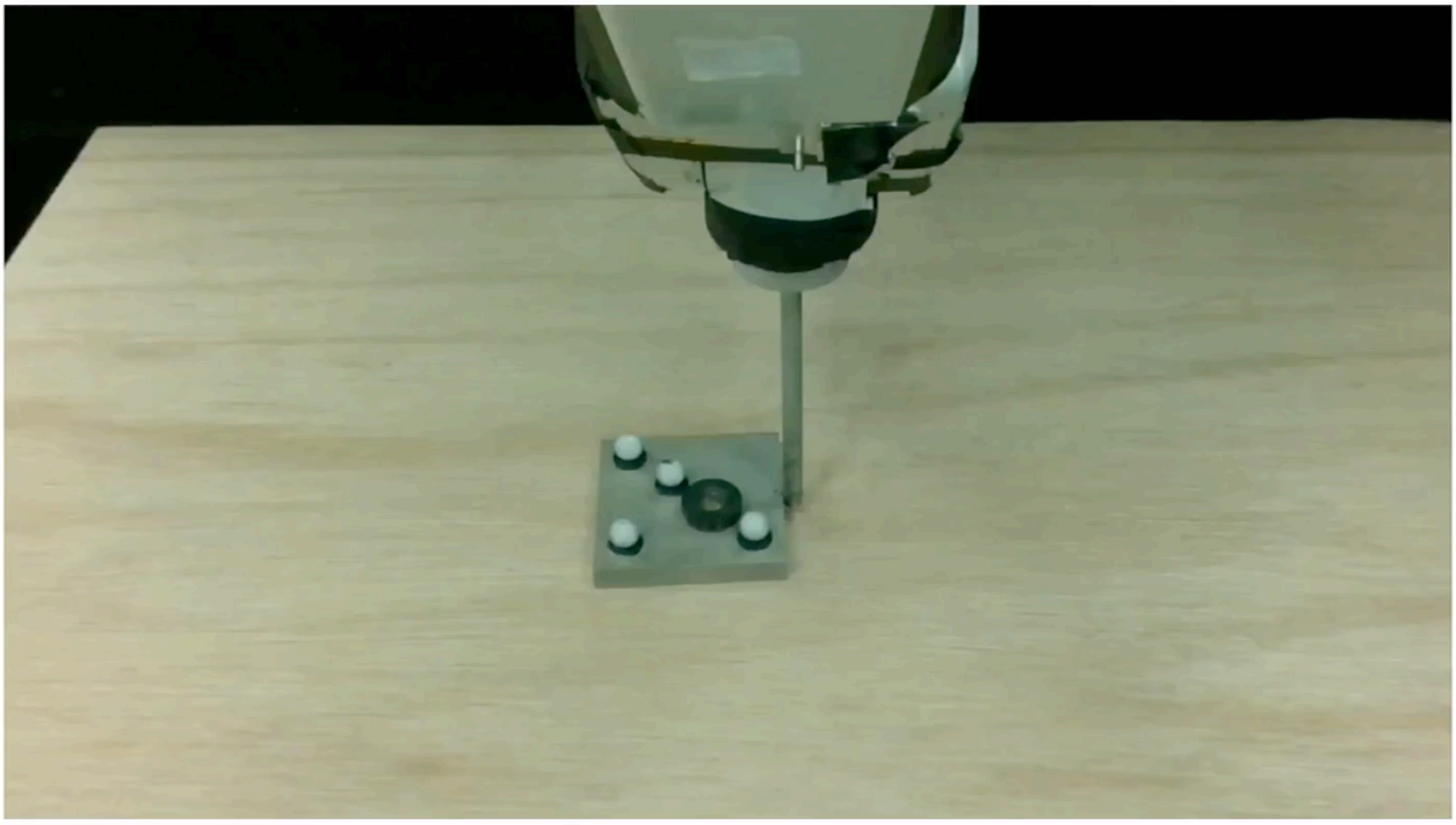




$$\text{NMSE} = \frac{\sum_{j=1}^{m}(y_j - \hat{y}_j)^2}{\sum_{j=1}^{m}(y_j - \bar{y})^2}$$

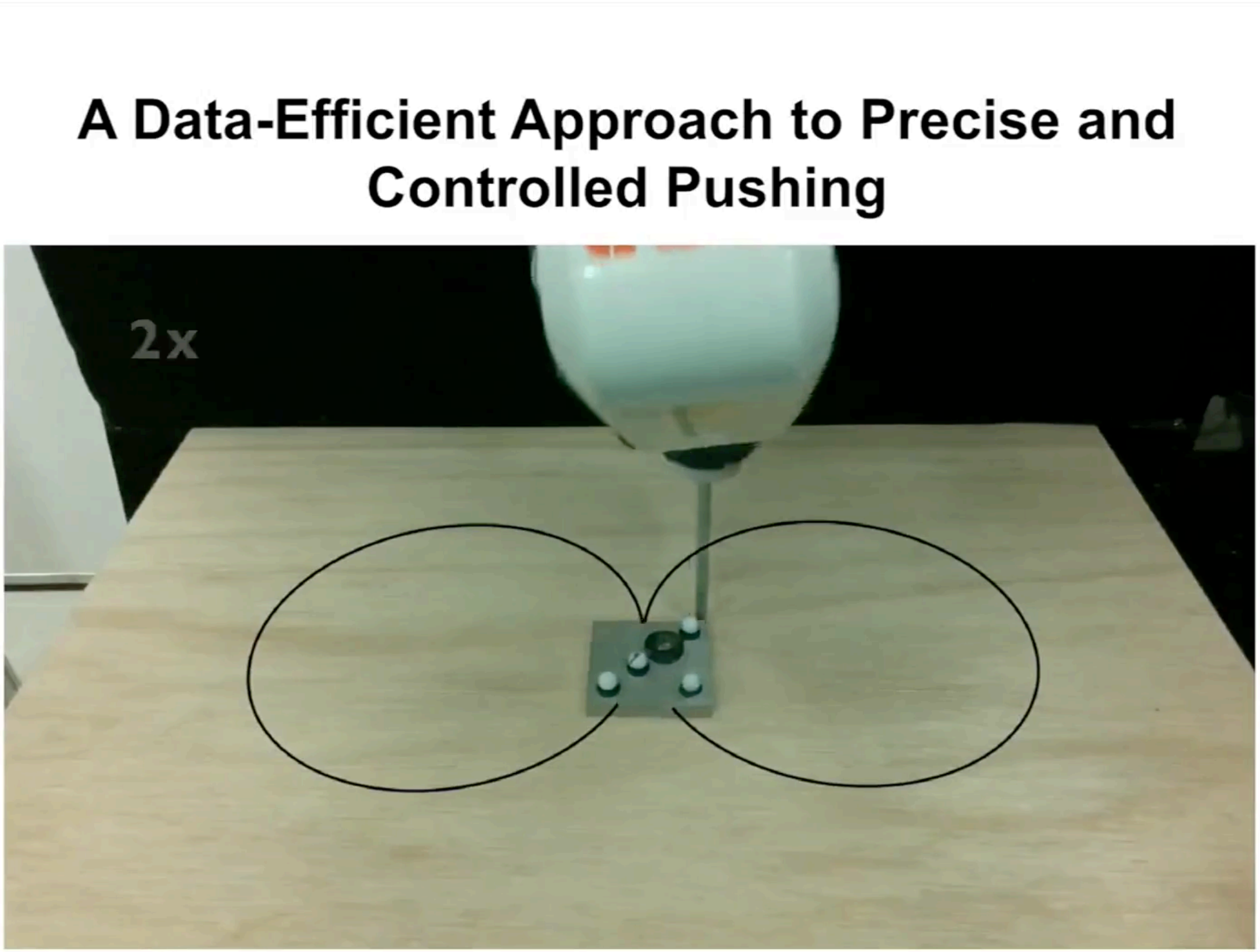


# Model for Pushing

## Data collection

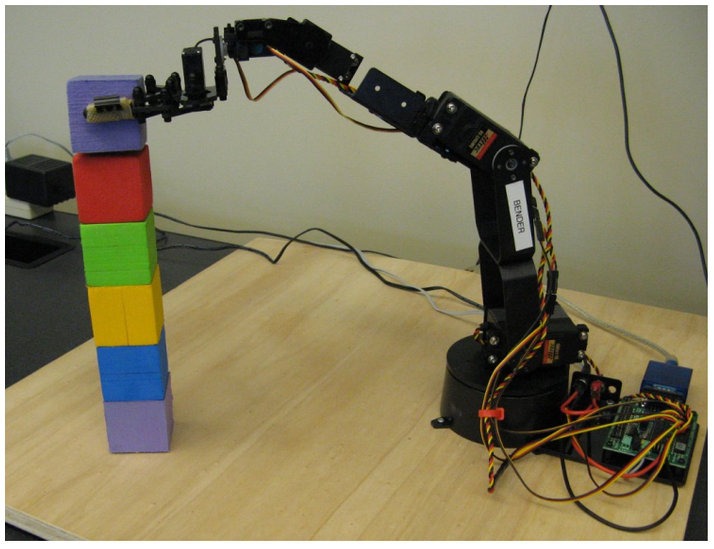


# Model for Pushing

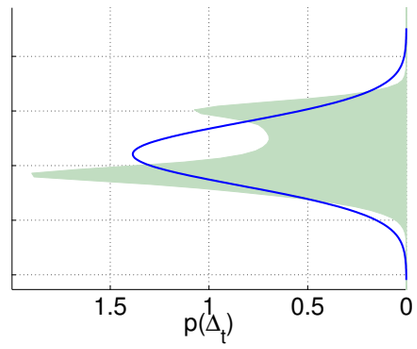


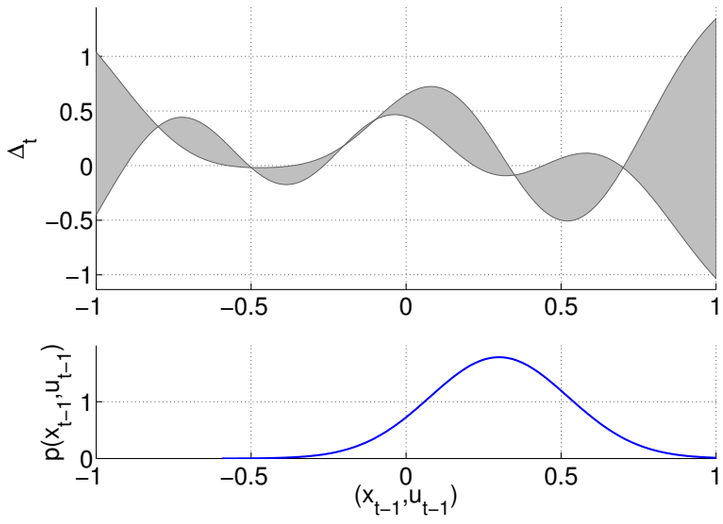
# Vision-based Block Stacking Example

- Learn (local) models to stack blocks
  - State (3D): centroid of block using blob tracking
  - Action (4D): pulse widths for first four joint servos



no-go area
obstacle cost

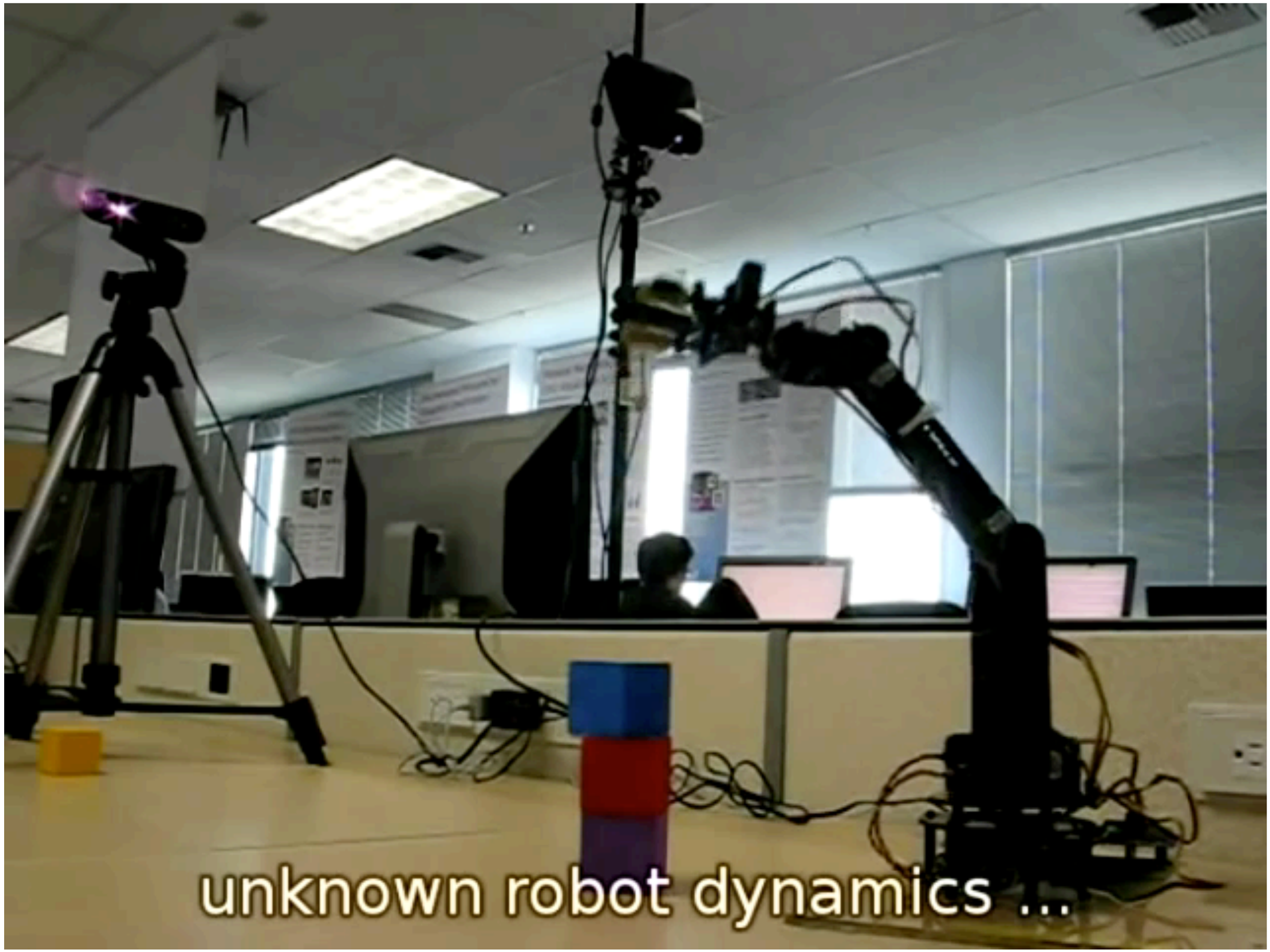



- Models can capture stochasticity/uncertainty
- Use model to compute an optimal policy

Deisenroth et al. Learning to Control a Low-Cost Manipulator using Data-Efficient Reinforcement Learning RSS 2011

# Vision-based Block Stacking Example



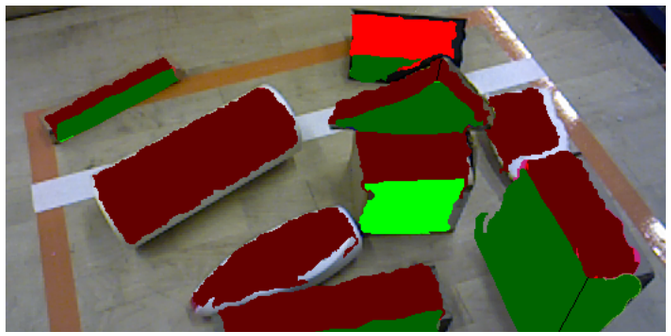

Deisenroth et al. Learning to Control a Low-Cost Manipulator using Data-Efficient Reinforcement Learning RSS 2011

# Affordances

# Affordances

- Objects afford actions/manipulations to an agent
  - Apple affords grasping, picking, eating, throwing… to a human
  - Apple affords grasping, picking, throwing… to a robot

**Surfaces that afford pushing to slide by a manipulator**

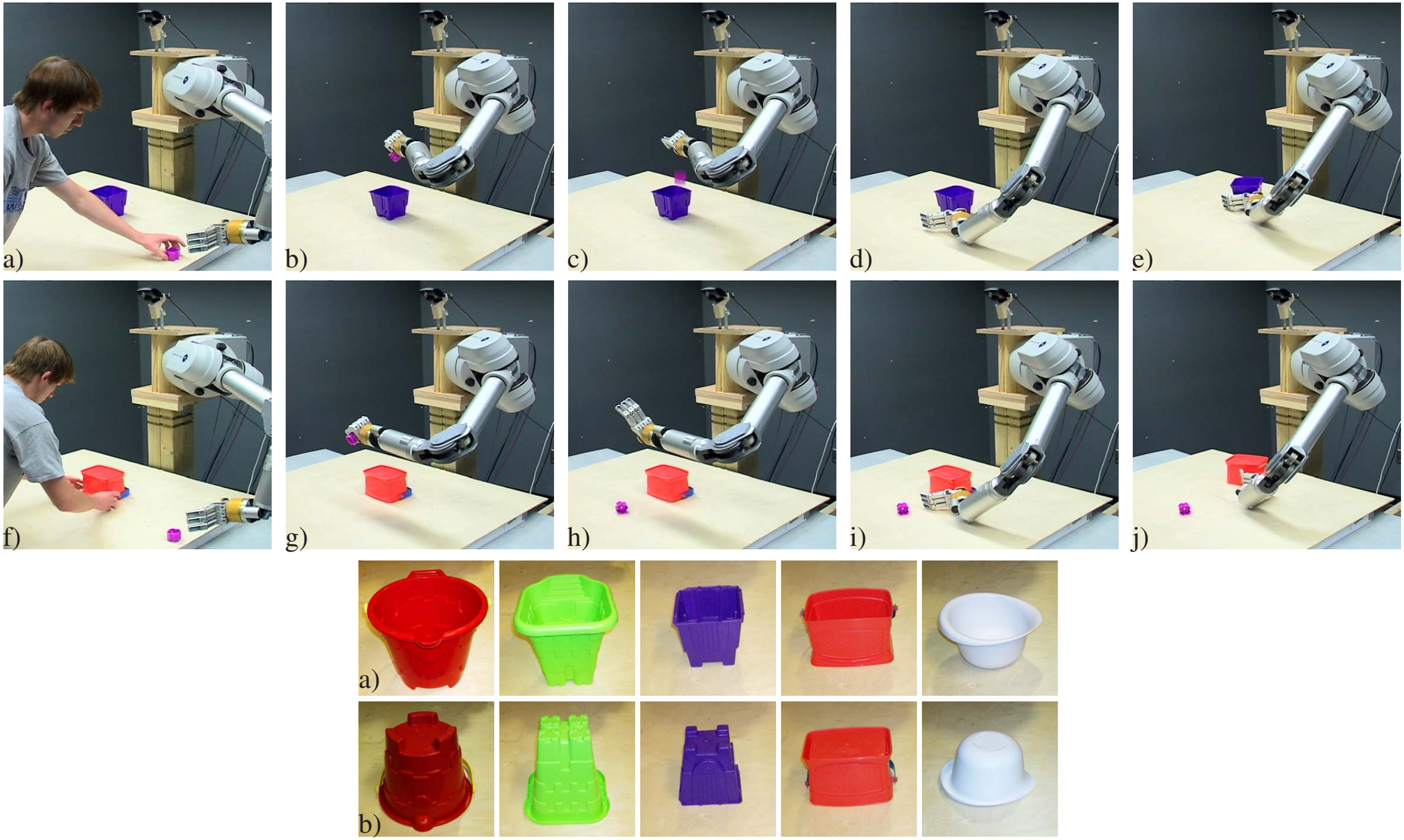


- Affordances connect objects and agents to action possibilities
  - Ground labels (graspable, liftable, etc.) in robot's actions
  - Robot can determine affordance label by performing action

Katz et al. Perceiving, Learning, and Exploiting Object Affordances for Autonomous Pile Manipulation, RSS 2013

# Container Example

learn from interactions with containers & non-containers



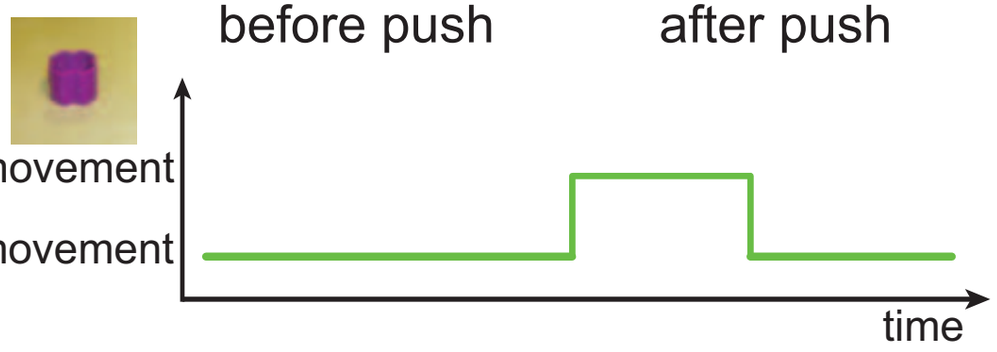

Griffith et al. Toward Interactive Learning of Object Categories by a Robot, ICDL 2009

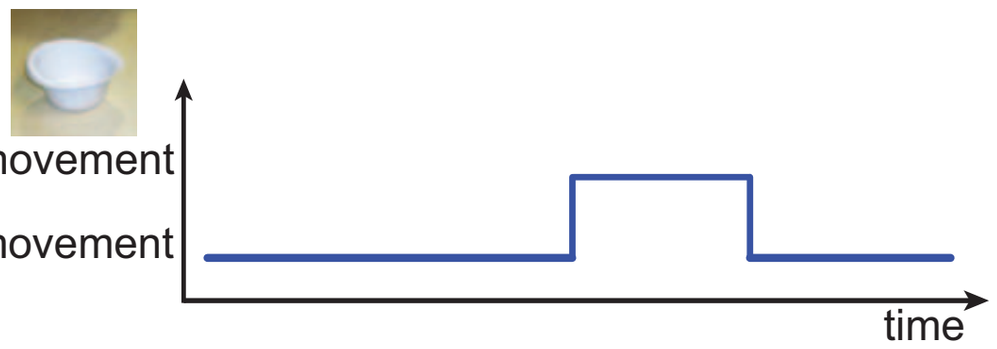
# Container Example

- Distinct classes of effects based on co-movement

Co-Movement

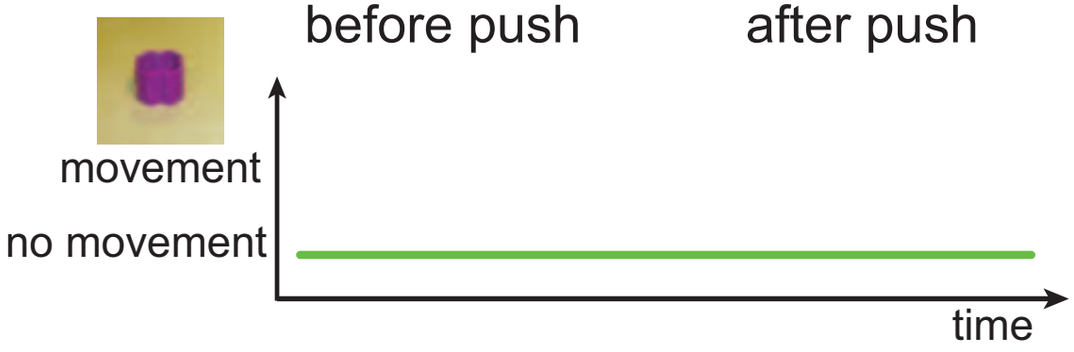


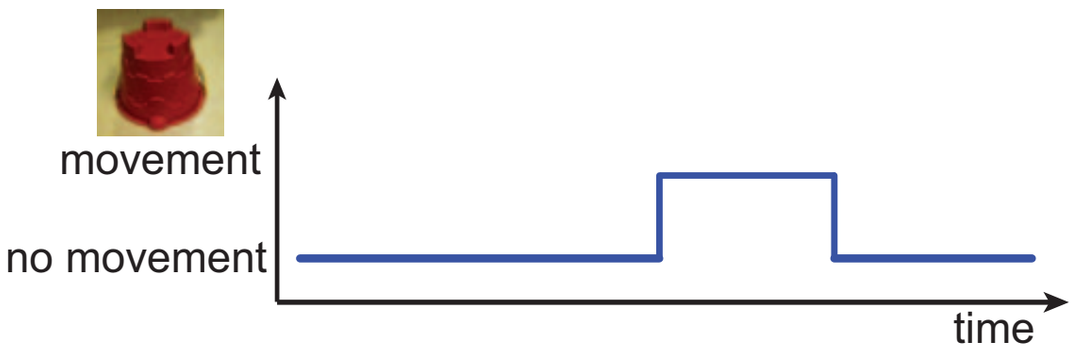



Separate Movement





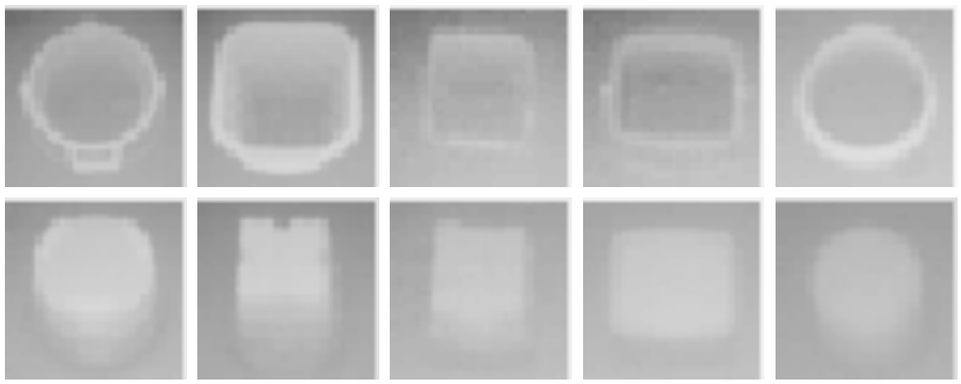




(a smaller third class corresponds to noisy tracking)

# Container Example

- Learn classifier for affordance (container or not)

Example Classifier Inputs



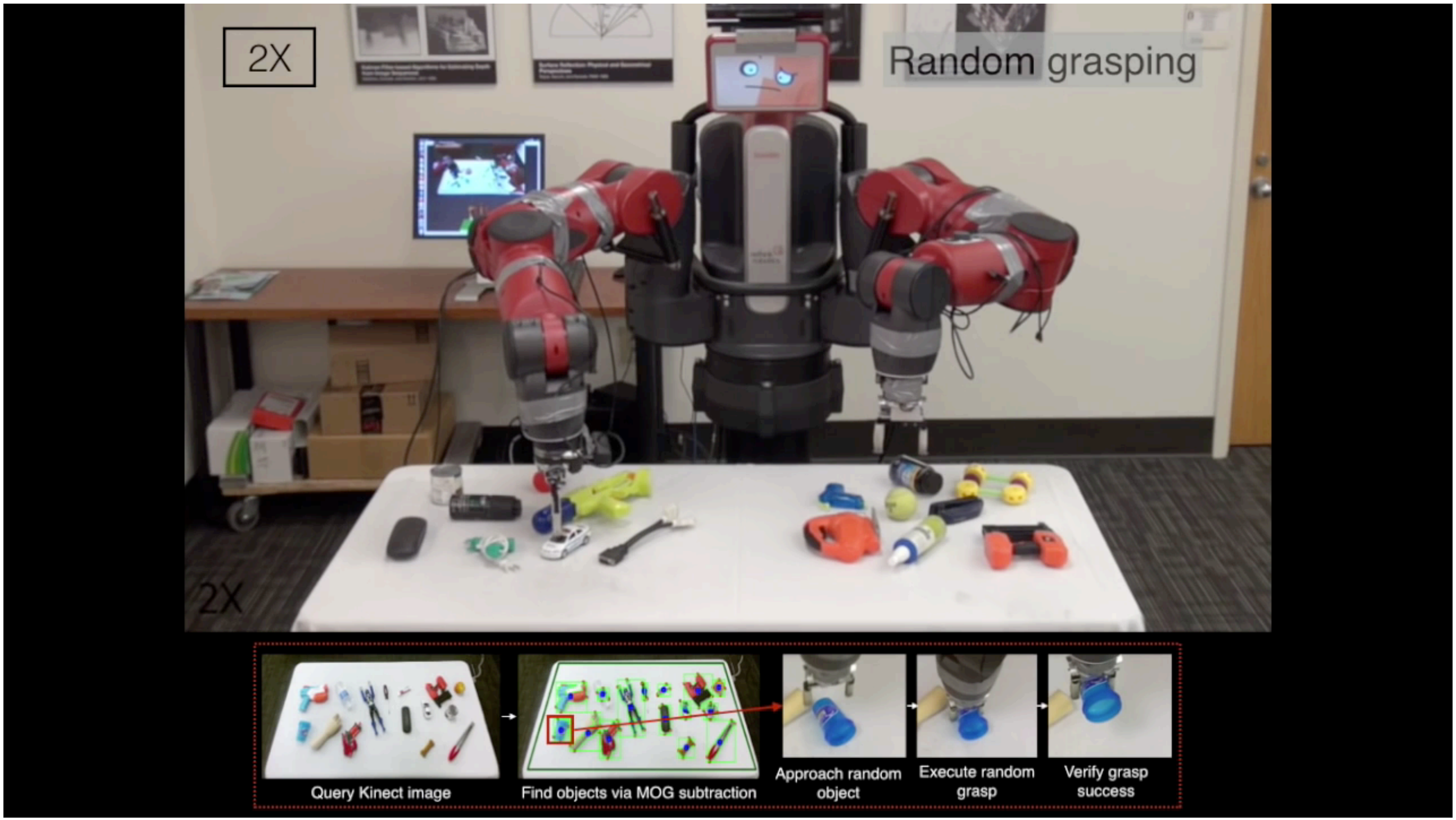
Classification Results

Novel Non-containers

Novel Containers

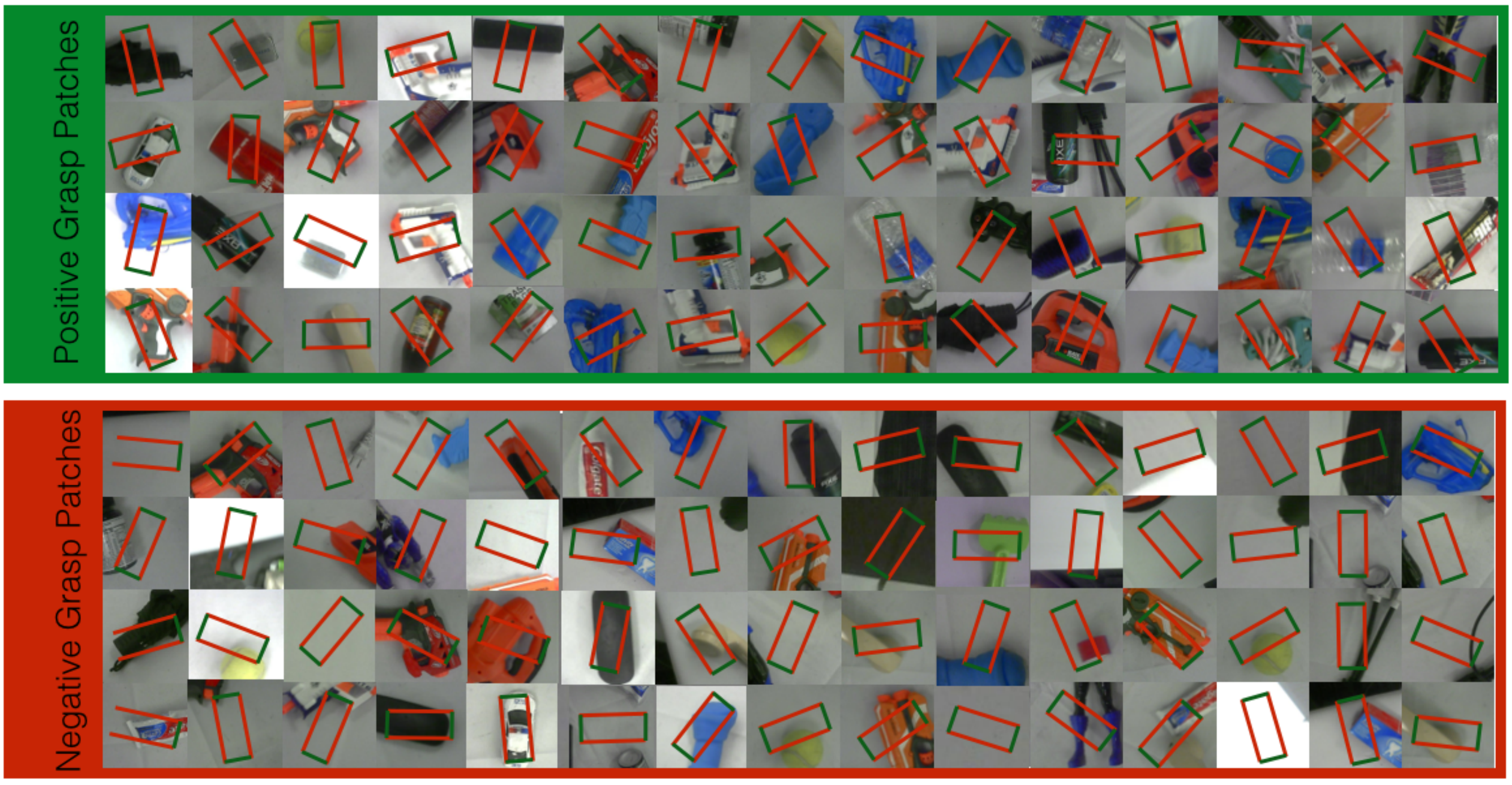
**Can learn container vs. non-container. What about cup vs. mug?**

# Self-supervised Grasp Affordances



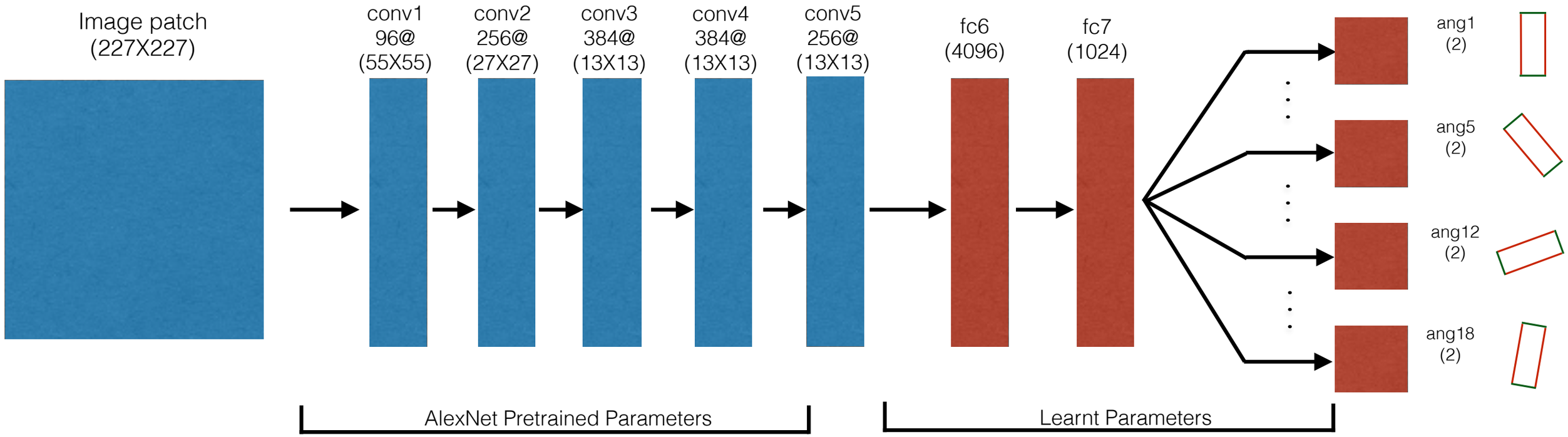

Pinto and Gupta, Supersizing Self-Supervision: Learning to Grasp from 50k tries and 700 Robot Hours, ICRA 2016

# Self-supervised Grasp Affordances




Pinto and Gupta, Supersizing Self-Supervision: Learning to Grasp from 50k tries and 700 Robot Hours, ICRA 2016

# Self-supervised Grasp Affordances



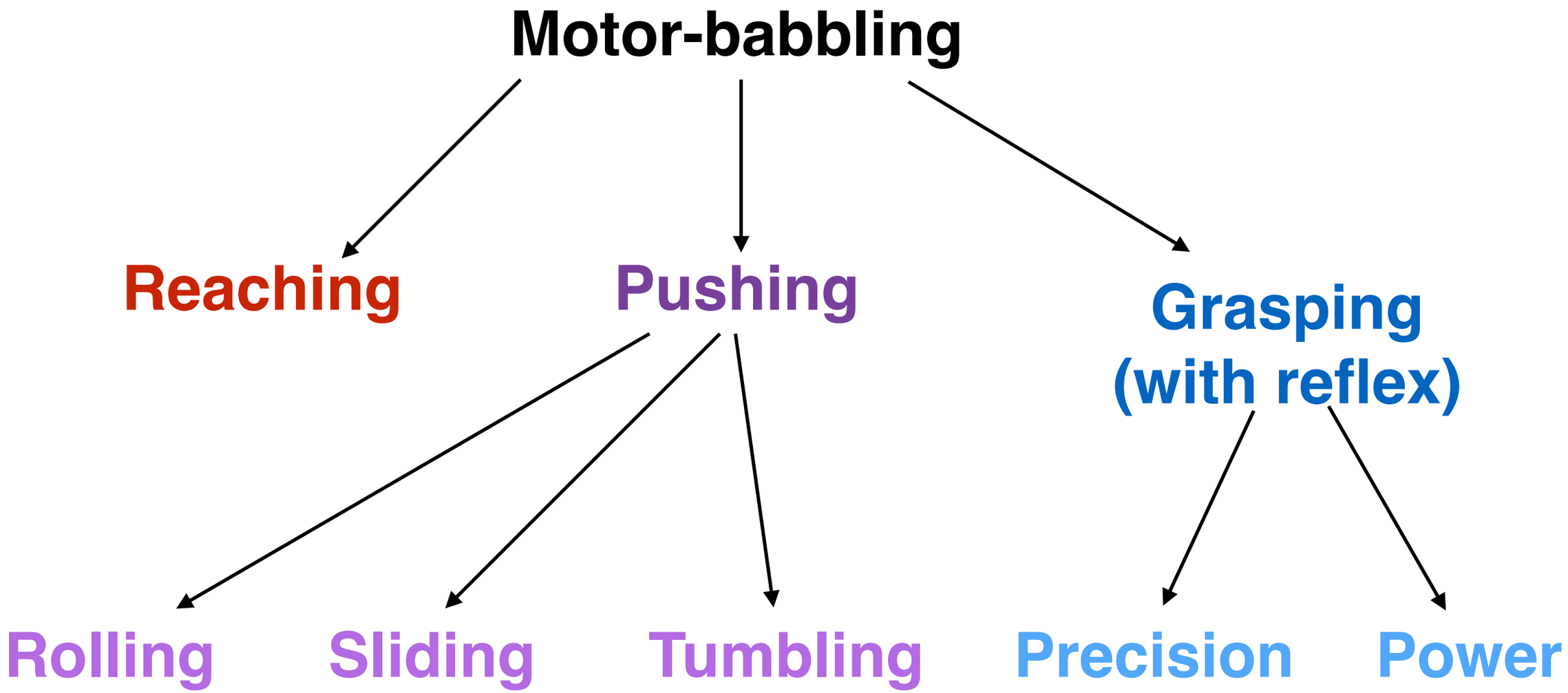

Pinto and Gupta, Supersizing Self-Supervision: Learning to Grasp from 50k tries and 700 Robot Hours, ICRA 2016

# Self-supervised Grasp Affordances


Pinto and Gupta, Supersizing Self-Supervision: Learning to Grasp from 50k tries and 700 Robot Hours, ICRA 2016

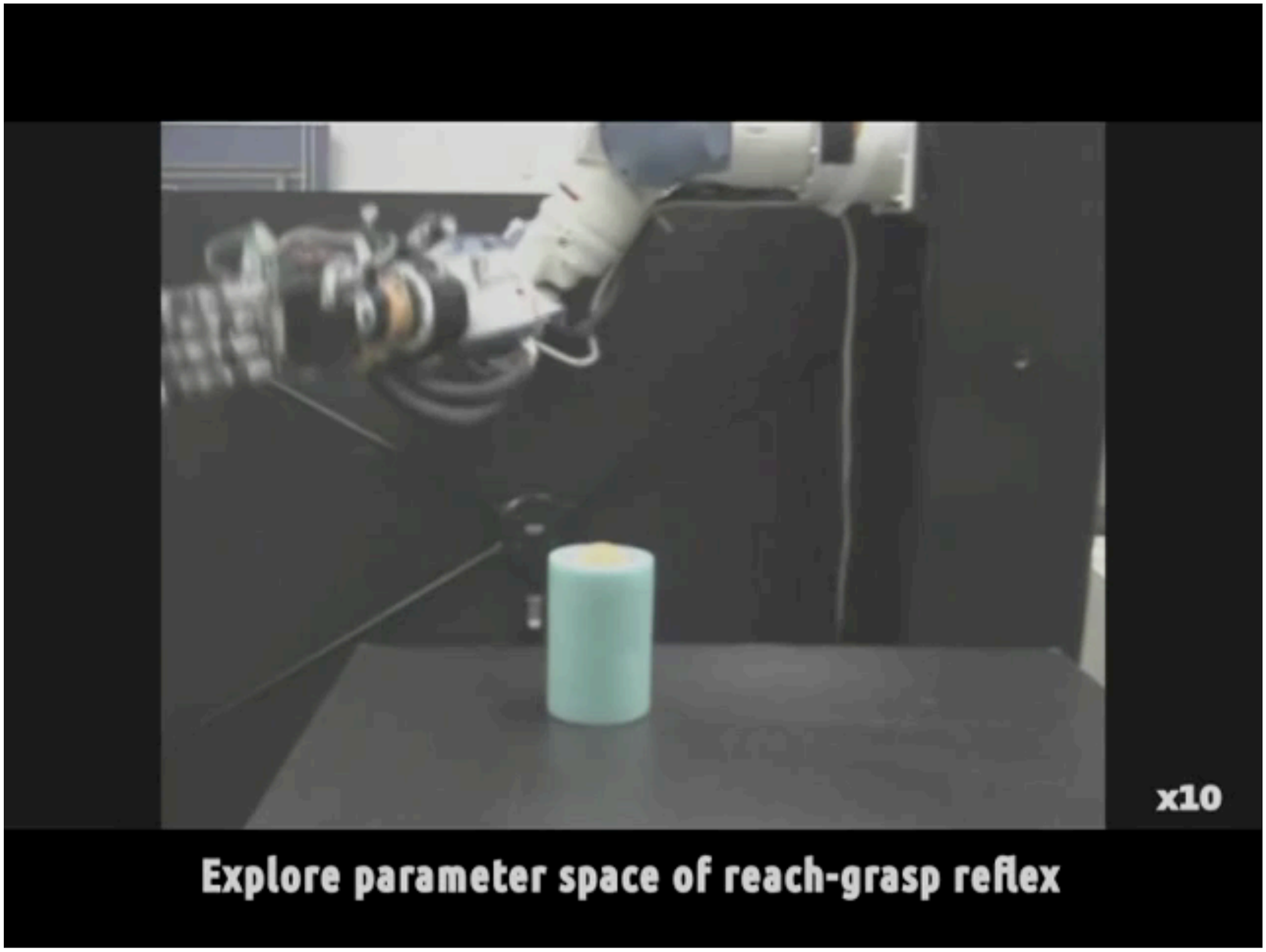
# Developmental Learning Example

- Interactions often lead to distinct salient effects
- Can cluster effects to create specific distinct skills




- Learn states that afford each skill (aka preconditions)

# Developmental Learning Example

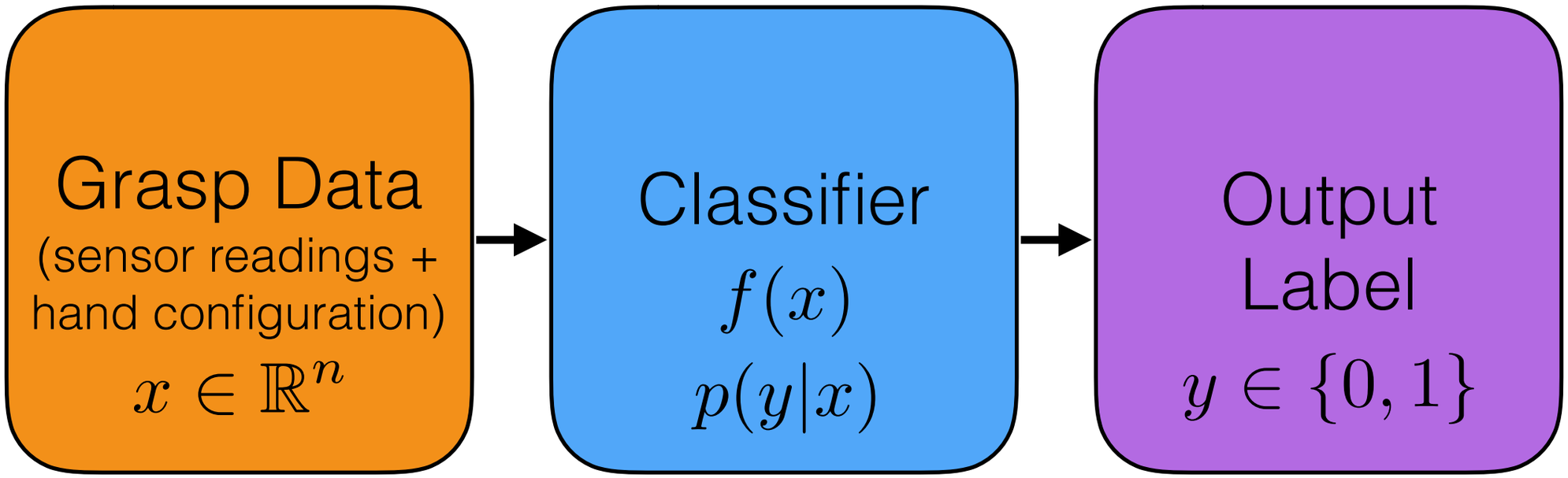



Ugur et al. Staged Development of Robotic Skills TAMD 2014

# Logistic Regression

# Classifying Grasps

- Classification problem from machine learning viewpoint




- Discriminant function defines decision boundary

$$f(x) = 0$$

- Probabilistic classifier models posterior distribution

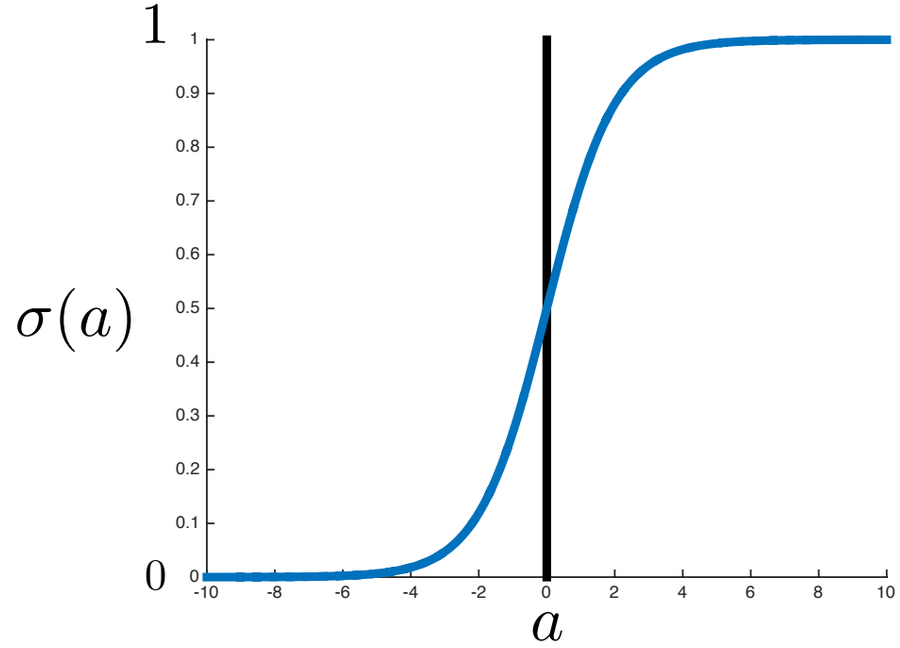
$$p(y|x)$$

# Logistic Regression

- Model posterior distribution using sigmoid function

$$p(y=1|x) = \sigma(\theta^T x) = \frac{1}{1+\exp(-\theta^T x)}$$

$$p(y=0|x) = 1 - \sigma(\theta^T x)$$




- Assume $x$ contains a constant bias term of 1
- Need to learn the parameters $\theta$ from training data

# Logistic Regression Training

- Training data consists of N labeled data points

$$y^{(1)}, y^{(2)}, ..., y^{(N)} \qquad\qquad x^{(1)}, x^{(2)}, ..., x^{(N)}$$

- Likelihood of the model parameters

$$L(\theta) = \prod_{i=1}^{N} \sigma(\theta^T x^{(i)})^{y^{(i)}} (1 - \sigma(\theta^T x^{(i)}))^{1-y^{(i)}}$$

- Negative log likelihood error function (cross entropy)

$$E(\theta) = -\log(L(\theta)) = -\sum_{i=1}^{N} y^{(i)} \log(\sigma(\theta^T x^{(i)})) + (1 - y^{(i)}) \log(1 - \sigma(\theta^T x^{(i)}))$$

- Optimize parameters using gradient decent

$$\nabla E(\theta) = \sum_{i=1}^{N} (\sigma(\theta^T x^{(i)}) - y^{(i)}) x^{(i)} \qquad\qquad \theta_{\text{new}} = \theta_{\text{old}} - \beta \nabla E(\theta_{\text{old}})$$

learning rate

# Logistic Regression Example

- Compute the updated parameters for the following data

$$x^{(1)} \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix}, y^{(1)} = 1 \qquad x^{(2)} \begin{bmatrix} 1 \\ 0.5 \\ 1 \end{bmatrix}, y^{(2)} = 0 \qquad x^{(3)} \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}, y^{(3)} = 1$$

$$\theta_{\text{old}} = \begin{bmatrix} 1 & 1 & -1 \end{bmatrix}^T, \beta = 0.1$$

$$x^{(1)} = \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix}, y^{(1)} = 1 \qquad x^{(2)} = \begin{bmatrix} 1 \\ 0.5 \\ 1 \end{bmatrix}, y^{(2)} = 0 \qquad x^{(3)} = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}, y^{(3)} = 1$$

---

$$\theta_{\text{old}} = \begin{bmatrix} 1 & 1 & -1 \end{bmatrix}^T, \beta = 0.1 \qquad \nabla E(\theta) = \sum_{i=1}^{N} (\sigma(\theta^T x^{(i)}) - y^{(i)}) x^{(i)}$$

- Compute current predictions

$$\sigma(0 + 2 - 1) = 0.7311 \qquad \sigma(1 + 0.5 - 1) = 0.6225 \qquad \sigma(-1 + 0 - 1) = 0.1192$$

- Compute errors

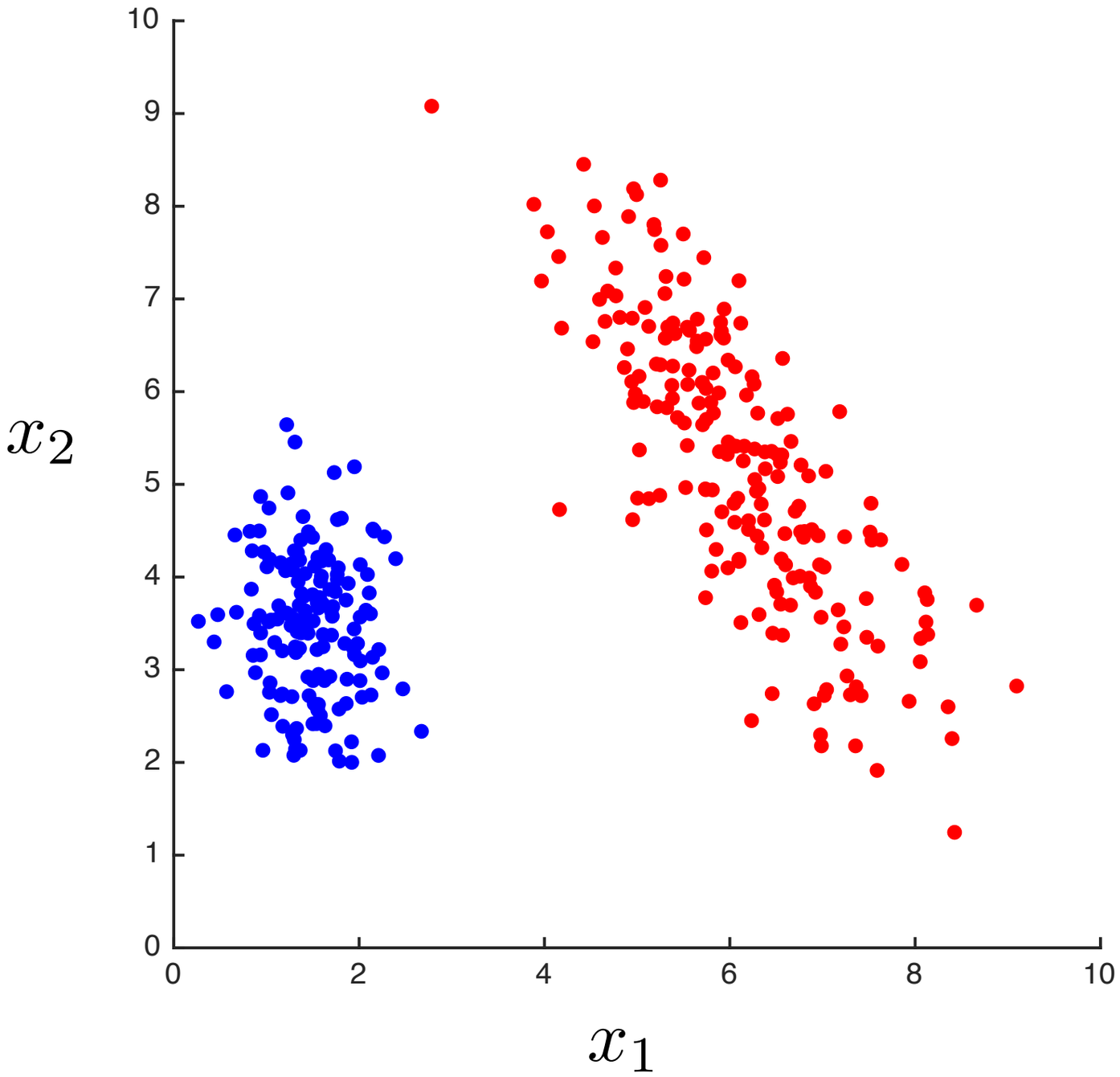
$$0.7311 - 1 = -0.2689 \qquad 0.6225 - 0 = 0.6225 \qquad 0.1192 - 1 = -0.8808$$

- Compute gradient and update parameters

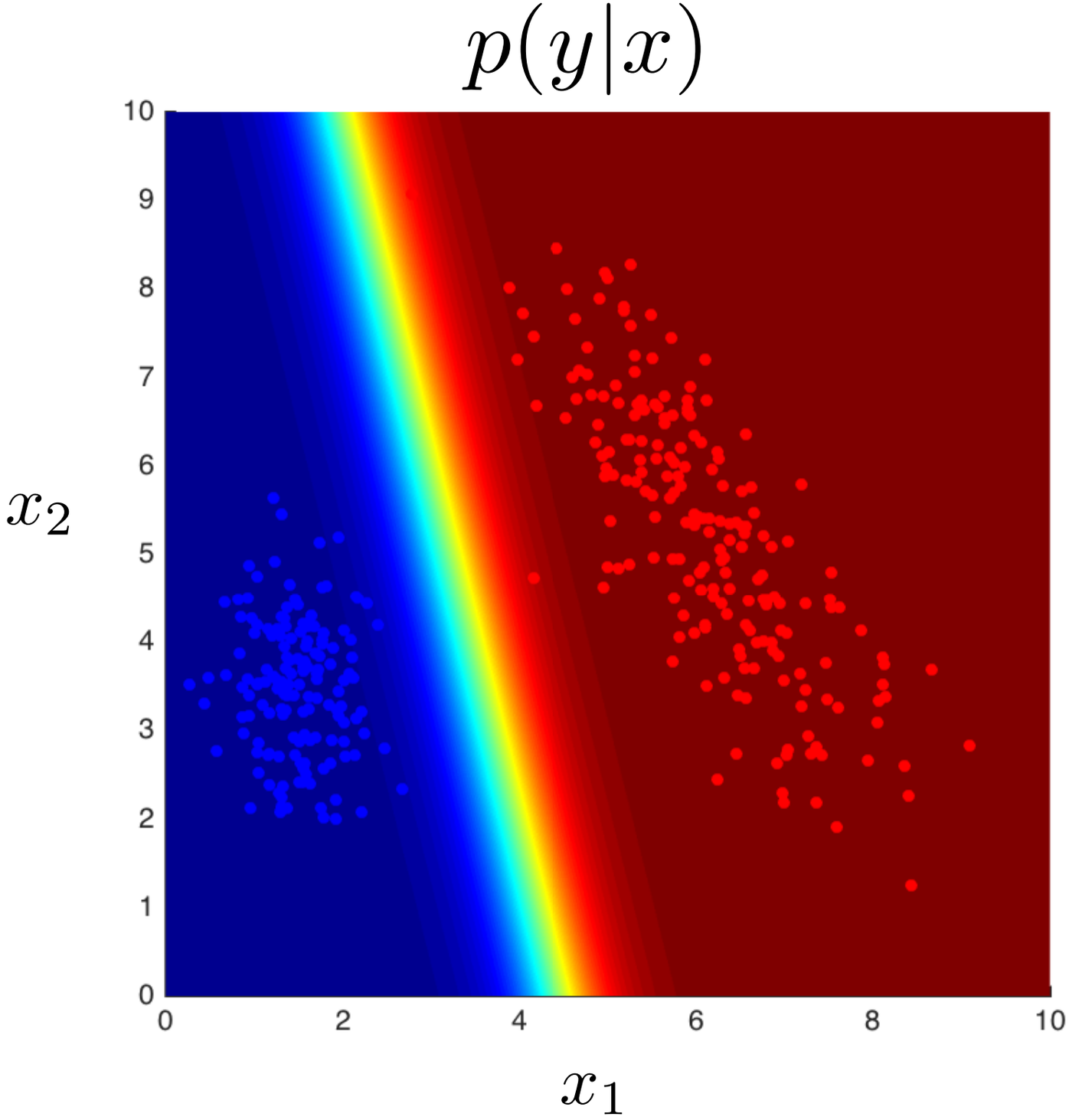
$$\nabla E = -0.2689 \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix} + 0.6225 \begin{bmatrix} 1 \\ 0.5 \\ 1 \end{bmatrix} - 0.8808 \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1.5033 \\ -0.2267 \\ -0.5273 \end{bmatrix}$$

$$\theta_{\text{new}} = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} - 0.1 \begin{bmatrix} 1.5033 \\ -0.2267 \\ -0.5273 \end{bmatrix} = \begin{bmatrix} 0.8497 \\ 1.0227 \\ -0.9473 \end{bmatrix}$$

# Logistic Regression Example

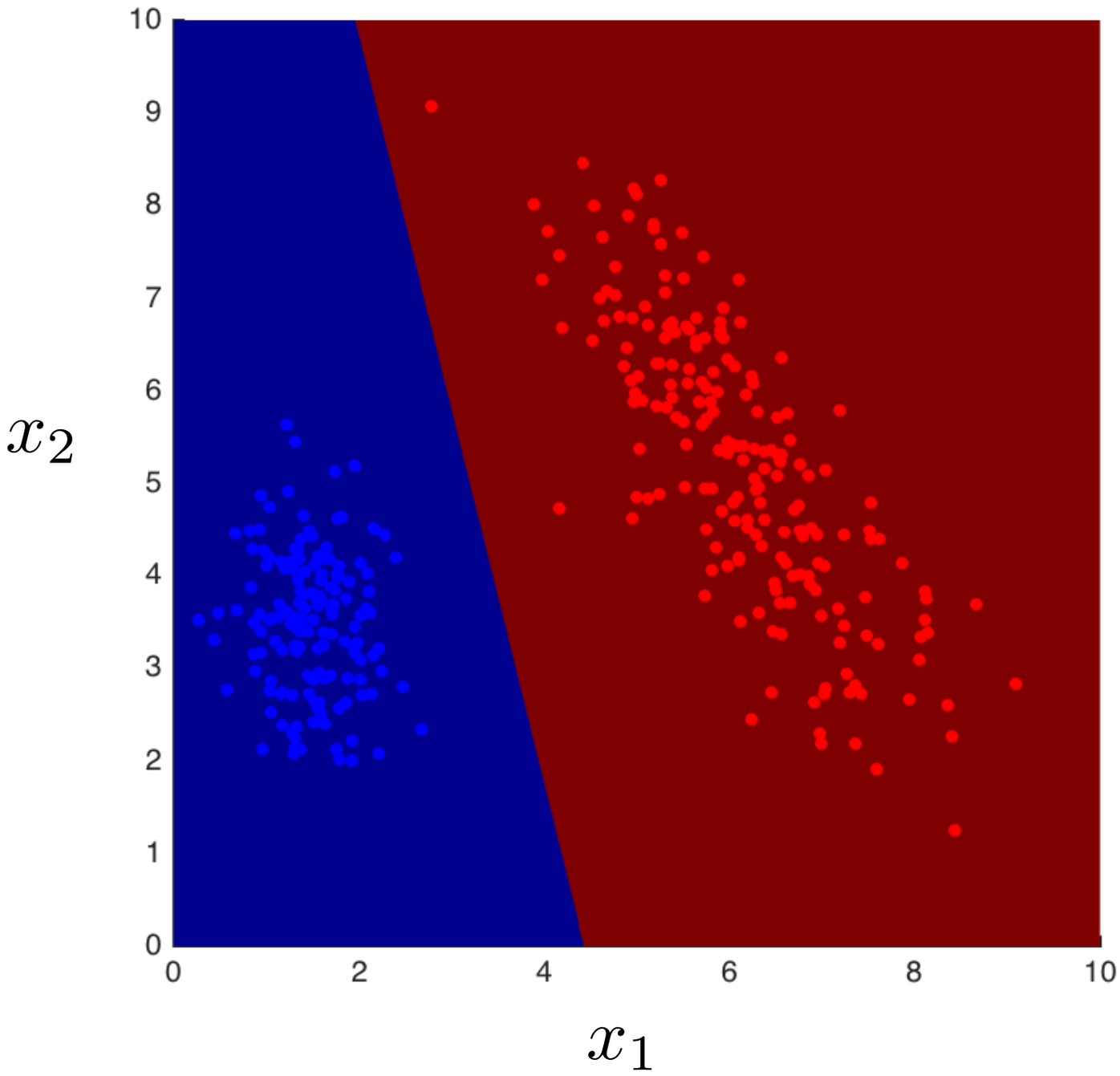


# Logistic Regression Example



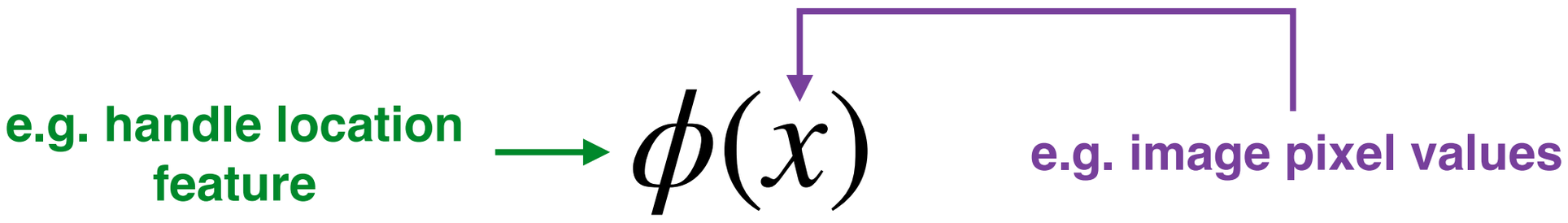
# Logistic Regression Example

## Decision Boundary



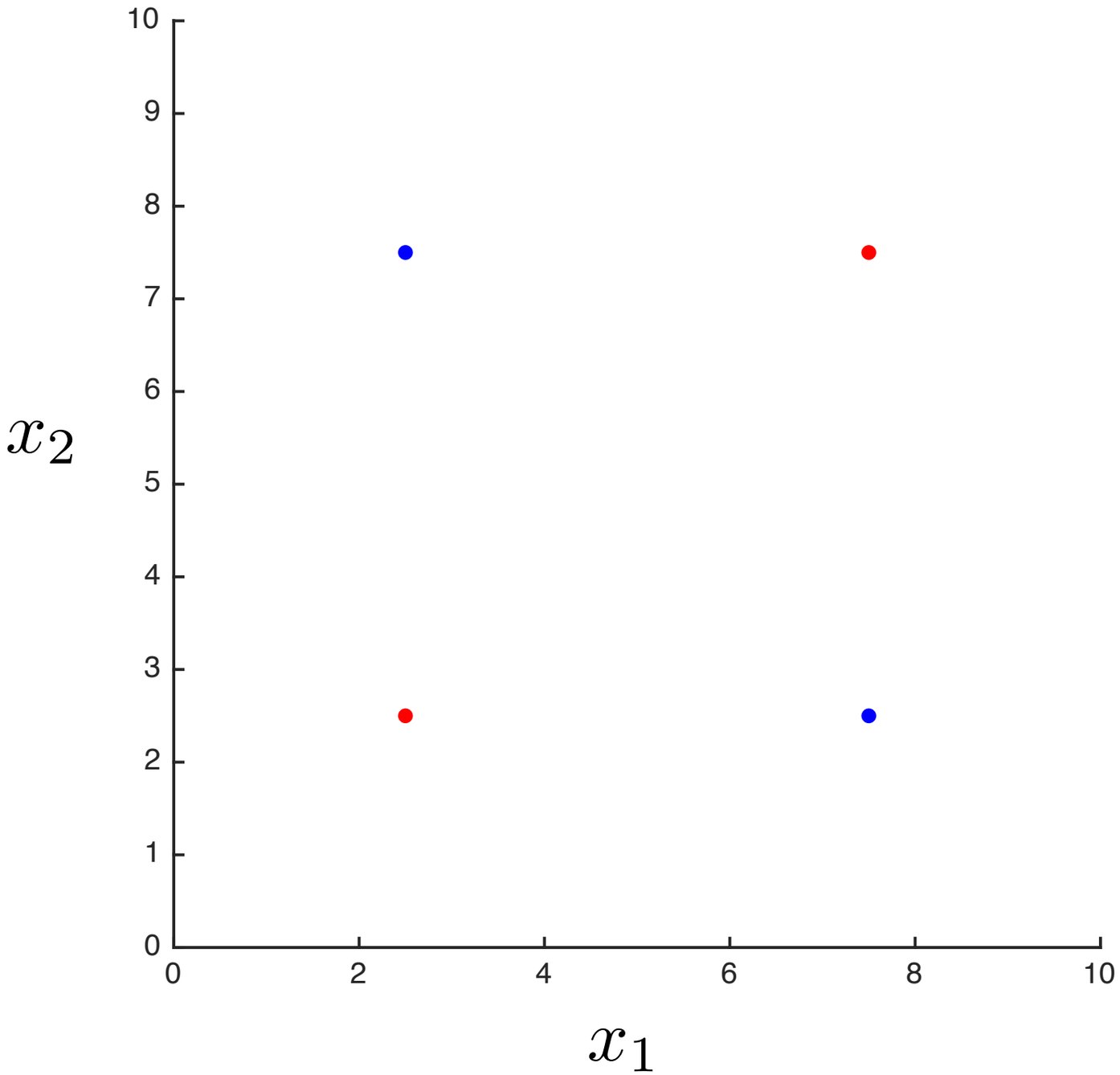
# Features

- A key challenge is finding a suitable input representation
- In practice we often talk about features of the input



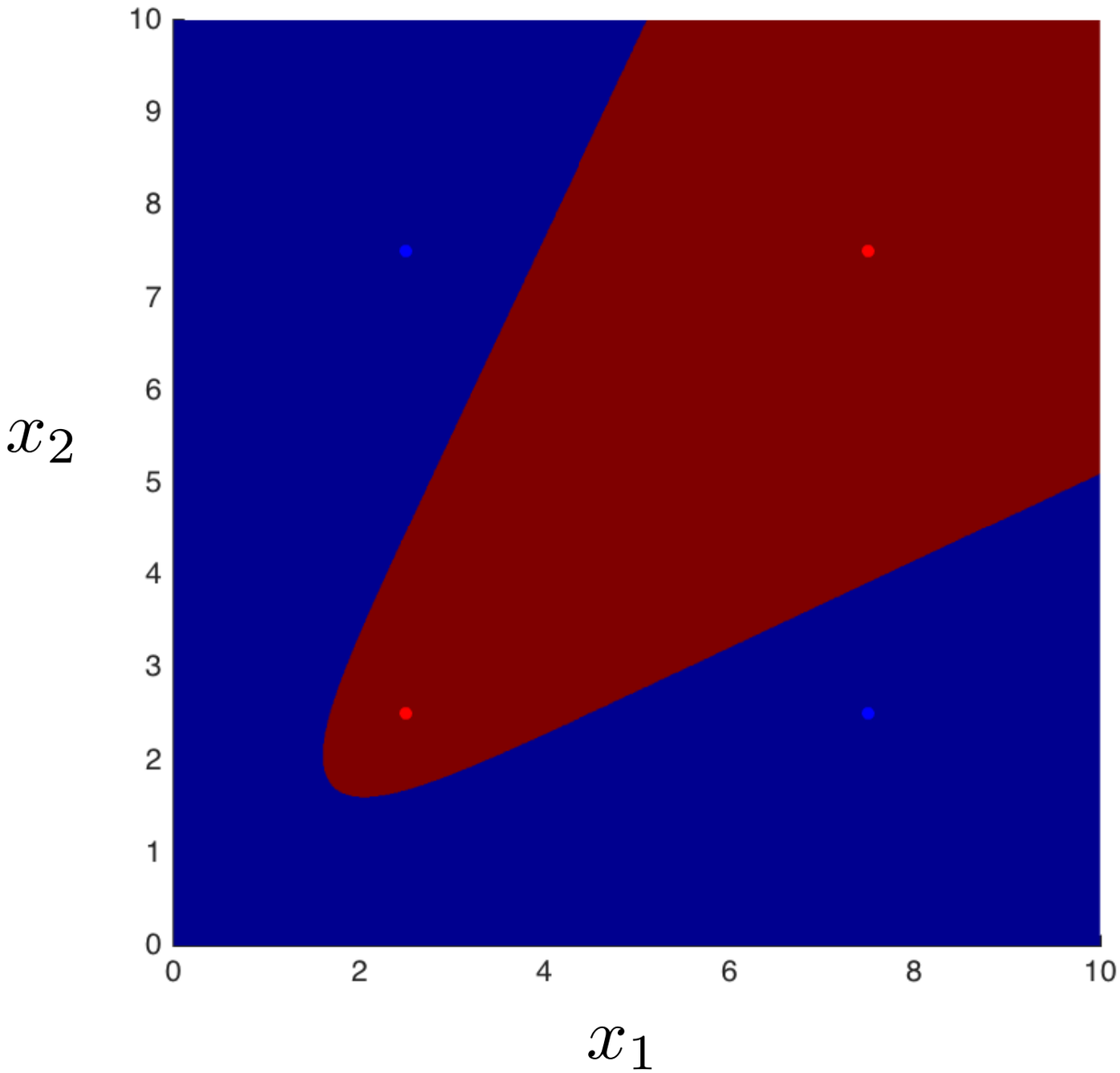

- Logistic regression same as before with $\phi(x)$ instead of $x$
  - Nonlinear features allow for nonlinear boundaries in x space
  - Neural networks can be seen as learning features

# Neural Network XOR Example



# Neural Network XOR Example

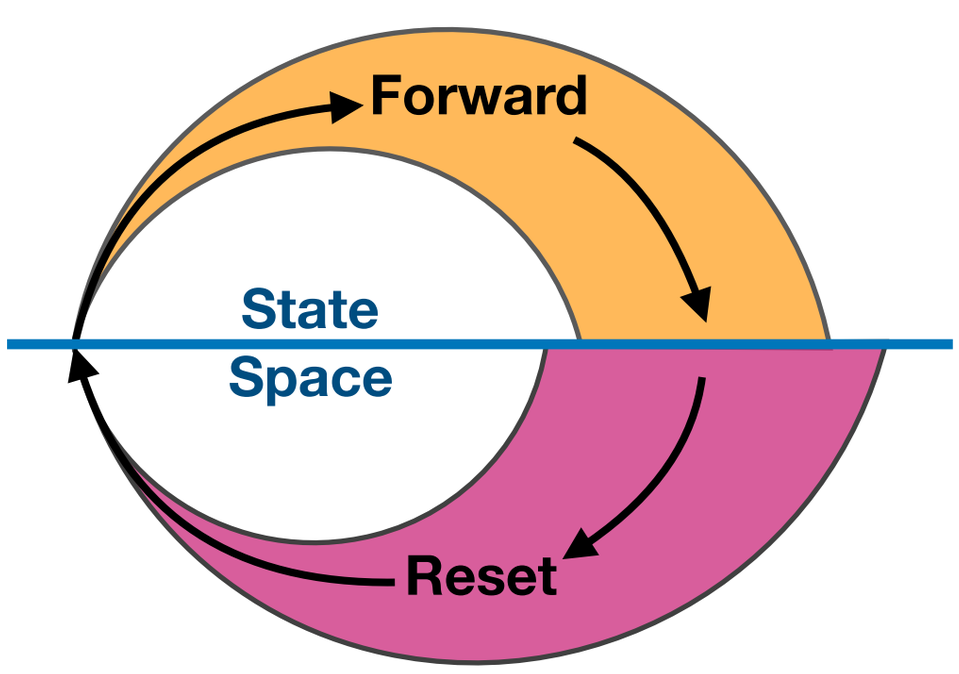
## Decision Boundary

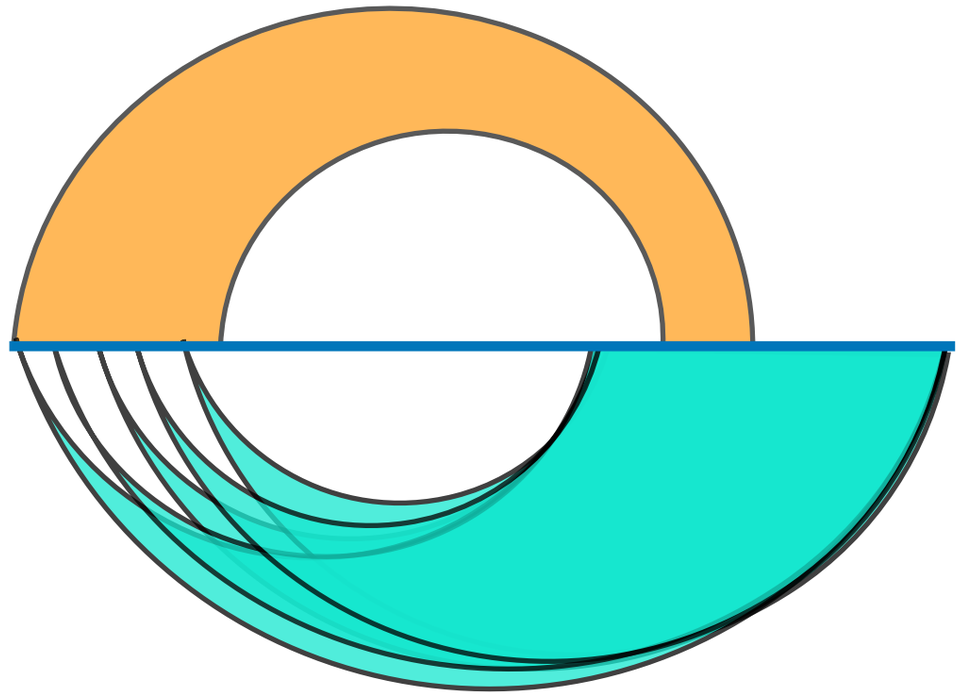


# Autonomous Reseting

# Resetting Environment

- Learning skills and models requires training data
- Often need to reset objects between trials

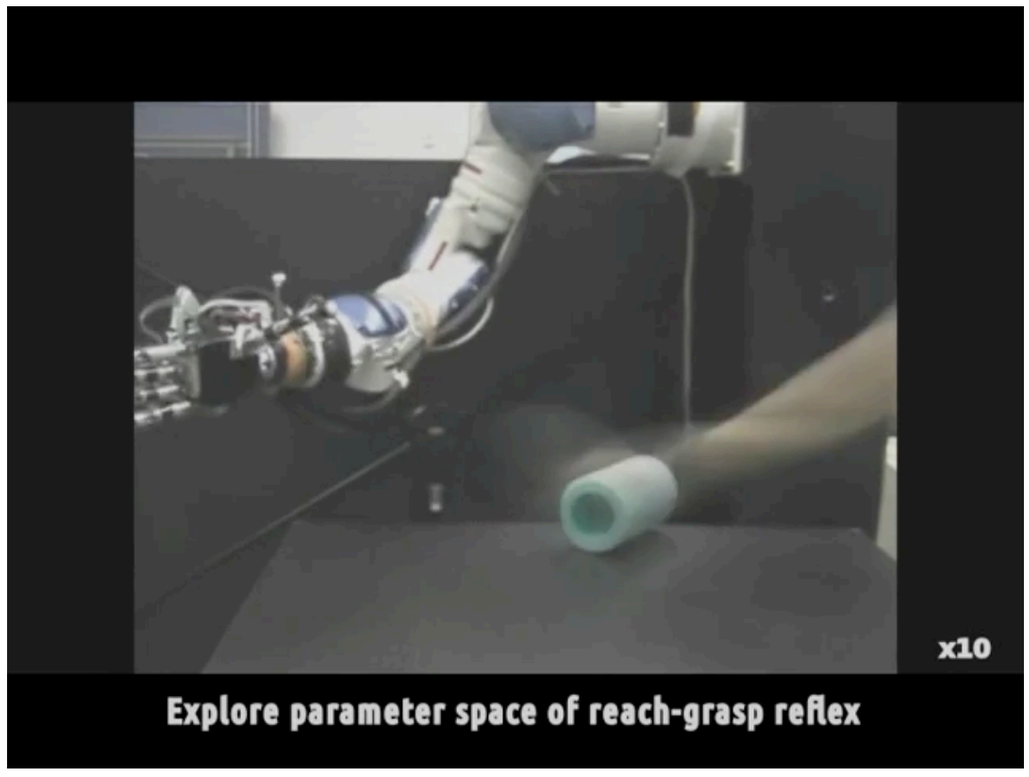


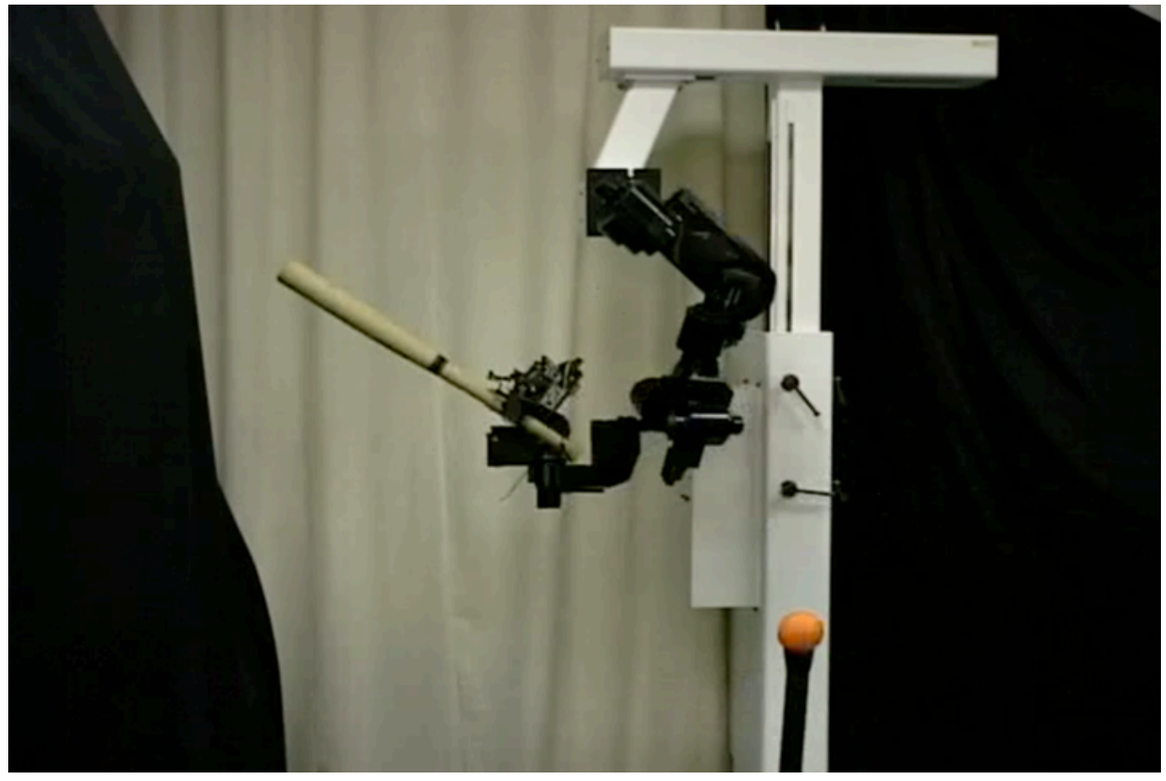



- Simulations allow for simple resetting
- Want to collect physical data from real world

# Human in the Loop

- Have a human reset the environment between trials



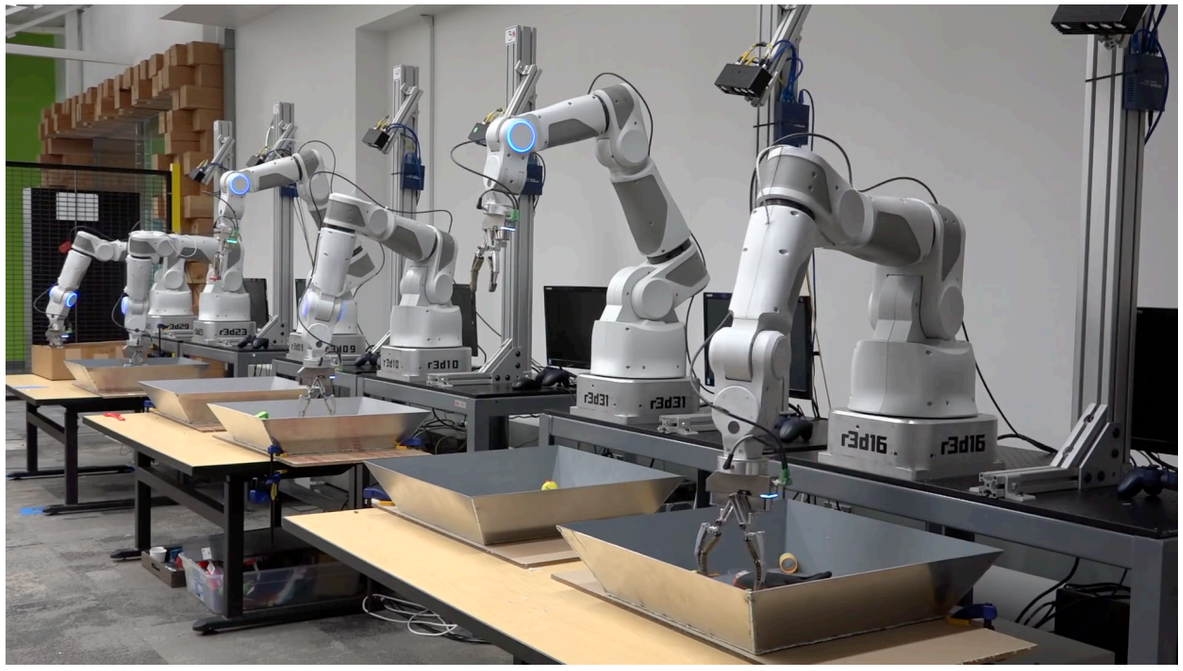
- Tedious and limits robot's overall autonomy
- Easy to explore a wide range of structured situations

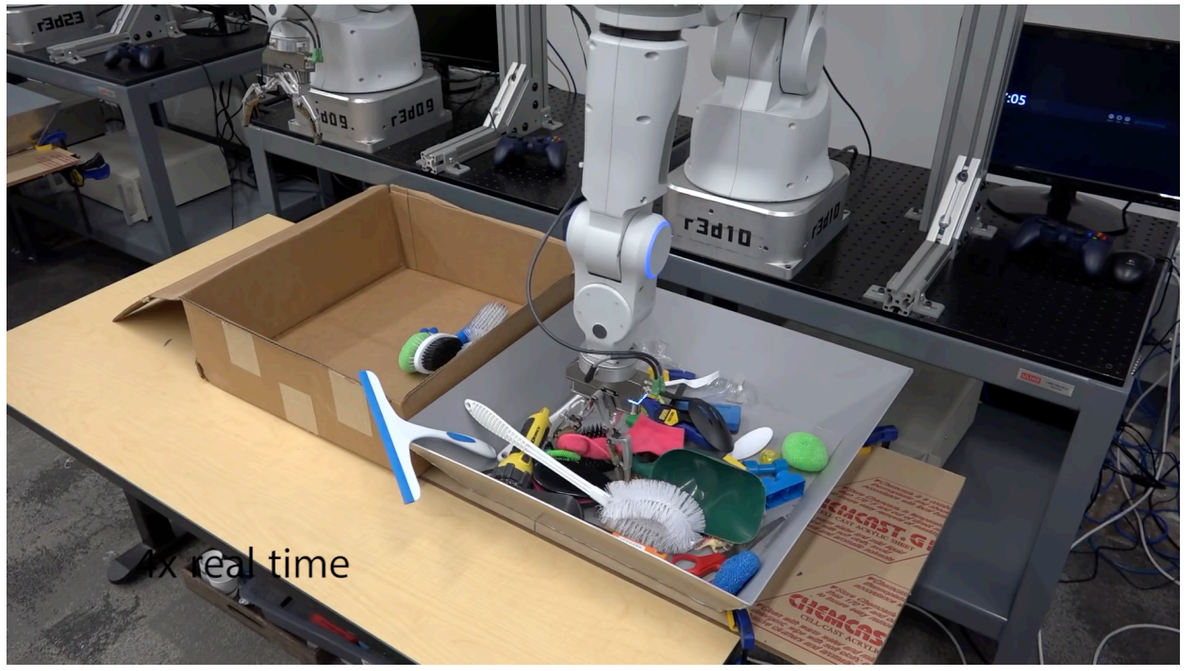
Ugur et al. Staged Development of Robotic Skills TAMD 2014

Peters et al. Natural Actor Critic, Neurocomputing 2008

# Working in Clutter

- Work in clutter to easily create diverse environments



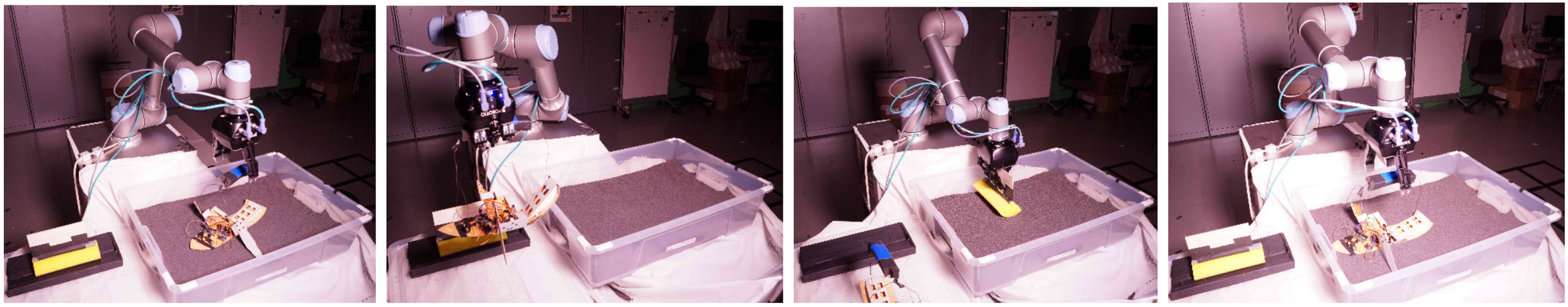



- Create new cluttered state by dropping objects
- Target rich environments for simple actions
- May not reflect real world scenes

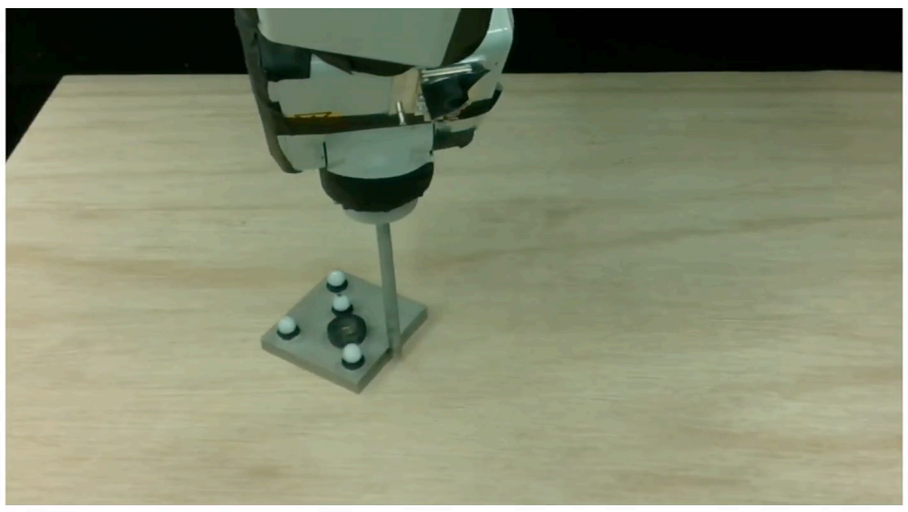
Levine et al., Learning Hand-Eye Coordination for Grasping with Deep Learning and Large-Scale Data Collection IJRR 2017

# Resetting Robot Actions

- Use robot to reset environment between trials



- Use pre-programmed actions to reset environment



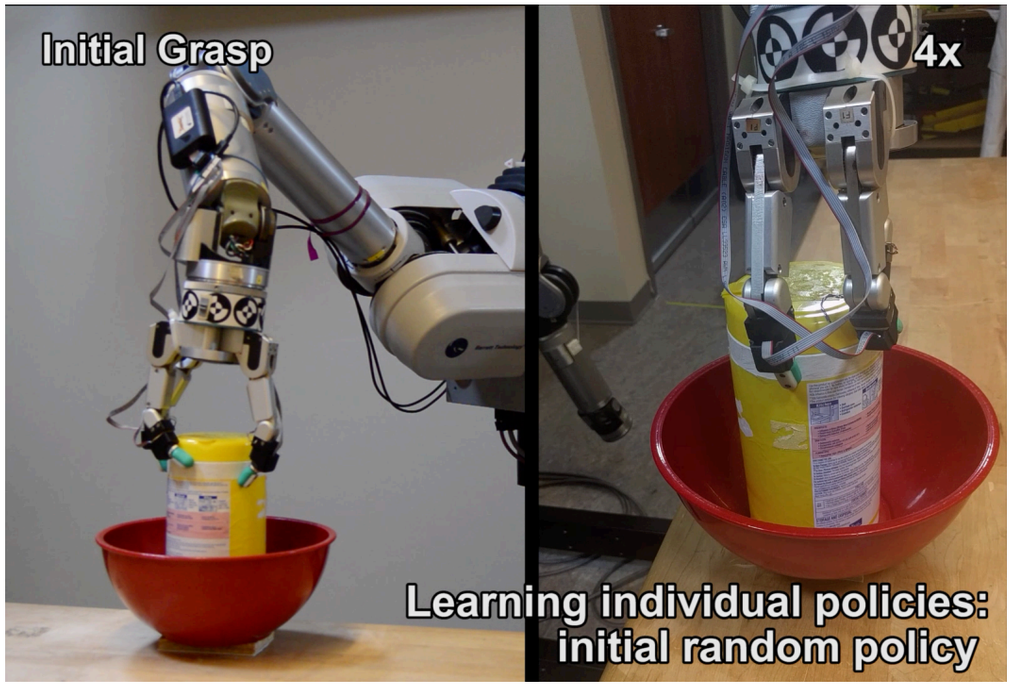
- Can augment objects to allow for more interactions, e.g., pulling

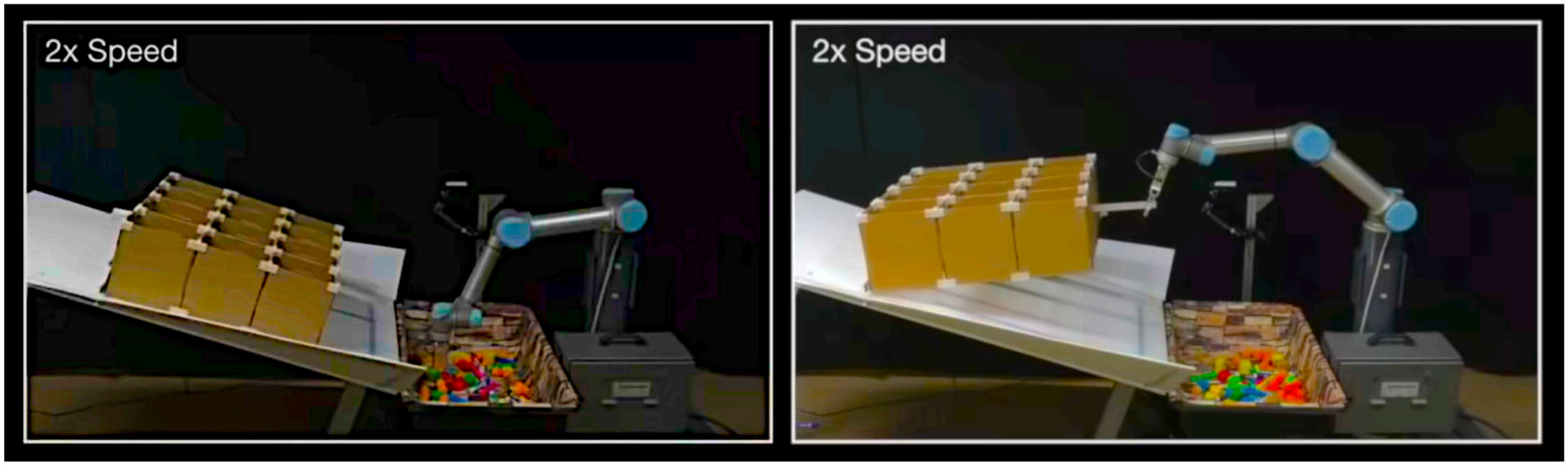
Kevin Luck et al. From the lab to the desert: Fast prototyping and learning of robot locomotion. Robotics: Science and Systems, 2017

Bauza and Rodriguez, A Probabilistic Data-Driven Model for Planar Pushing ICRA 2017

# Resetting Mechanisms

- Create structures in environment to help reset trials




Zeng et al Tossingbot: Learning to throw arbitrary objects with residual physics. RSS, 2019.
Chebotar et al. Generalizing regrasping with supervised policy learning. ISER, 2016.

# Summary

- Robots can learn models from experience
  - Perform actions and observe the resulting effects
- Interactive perception
  - Observe latent information by interacting with objects
- Transition models
  - Predict the effects of different actions
- Affordances
  - Determine if an object/state affords a desired skill
- Autonomous resetting allows for easier data collection

# Questions?